Click to verify



```
Point 1 - Install and run There should be no problems here, because here the process is similar to installing a regular core for a server or a bungee. 1) Download Velocity from the Papermc website. 2) Put the one you downloaded .jar file to a specific folder, then create a file in it start.bat (start.sh if linux), in which write the following: java -Xmx2G
Xms2G -XX:+UseG1GC -XX:G1HeapRegionSize=4M -XX:+UnlockExperimentalVMOptions -XX:+ParallelRefProcEnabled -XX:+AlwaysPreTouch -XX:MaxInlineLevel=15 -jar velocity.jar (Note. You will need Java version 11 or higher!) 3) Now run it. After that, wait until all folders and files are created and after a message like "Done (2,2s)!" close the
console window that opens. Run it again after full configuration. Point 2 - Velocity setting Velocity has one configuration file - velocity setting Velocity setting Velocity has one configuration file - velocity setting Velocit
connections = false # Should we forward IP addresses and other data to backend servers? # Available options: # - "none": No forwarding will be done. All players will appear to be connecting # from the proxy and will have offline-mode UUIDs. # - "legacy": Forward player IPs and UUIDs in a BungeeCord-compatible format. Use this # if you run
servers using Minecraft 1.12 or lower. # - "bungeeguard": Forward player IPs and UUIDs in a format supported by the BungeeGuard # plugin. Use this if you run servers using Minecraft 1.12 or lower, and are # unable to implement network level firewalling (on a shared host). # - "modern": Forward player IPs and UUIDs as part of the login process
using # Velocity's native forwarding. Only applicable for Minecraft 1.13 or higher. player-info-forwarding.secret # If you are using modern or BungeeGuard IP forwarding.secret file = "forwarding.secret" # Announce
whether or not your server supports Forge. If you run a modded server, we # suggest turning this on. # # If your network runs one modpack consistently, consider using ping-passthrough = "mods" # instead for a nicer display in the server list. announce-forge = false # If enabled (default is false) and the proxy is in online mode, Velocity will kick #
any existing player who is online if a duplicate connection attempt is made. kick-existing-players = false # Should Velocity pass server list ping response. # - "mods": Passes
only the mod list from your backend server into the response. # The first server in your try list (or forced host) with a mod list will be # used. If no backend server can be contacted, Velocity won't display any # mod information. # - "description": Uses the description and mod list from the backend server. The first # server in the try (or forced host)
list that responds is used for the # description and mod list. # - "all": Uses the backend server's response as the proxy response as the proxy response. The Velocity # configuration is used if no servers could be contacted by in logs enable-player-address-logging = true
[servers] # Configure your servers here. Each key represents the server's name, and the value # represents the IP address of the server to connect to. lobby = "127.0.0.1:30066" factions = "1
hosts here. "lobby.example.com" = [ "lobby" ] "factions.example.com" = [ "lobby ] "factions.example.c
unexpected-server-disconnect = true announce-proxy-commands = true log-command-executions = false log-player-connections = true [query] enabled = false port = 25577 map = "Velocity" show-plugins = false log-player-connections = true [query] enabled = false port = 25577 map = "Velocity" show-plugins = false log-player-connections = true [query] enabled = false port = 25577 map = "Velocity" show-plugins = false log-player-connections = true [query] enabled = false port = 25577 map = "Velocity" show-plugins = false log-player-connections = true [query] enabled = false port = 25577 map = "Velocity" show-plugins = false log-player-connections = true [query] enabled = false port = 25577 map = "Velocity" show-plugins = false log-player-connections = true [query] enabled = false port = 25577 map = "Velocity" show-plugins = false log-player-connections = true [query] enabled = false port = 25577 map = "Velocity" show-plugins = false log-player-connections = true [query] enabled = false port = 25577 map = "Velocity" show-plugins = false log-player-connections = true [query] enabled = false port = 25577 map = "Velocity" show-plugins = false log-player-connections = true [query] enabled = false port = 25577 map = "Velocity" show-plugins = false log-player-connections = true [query] enabled = false port = 25577 map = "Velocity" show-plugins = false log-player-connections = true [query] enabled = false port = 25577 map = "Velocity" show-plugins = false log-player-connections = true [query] enabled = false port = 25577 map = "Velocity" show-plugins = false log-player-connections = true [query] enabled = false port = 25577 map = "Velocity" show-plugins = false log-player-connections = false l
server on versions 1.13+, then as indicated, use the player-info-forwarding-mode = "bungeeguard". Specify this value according to what is suitable for you. Next, we need to come up with a key and specify it in the file that is
specified in the forwarding-secret = "forwarding secret" column. If you chosen modern mode, then this key will need to be set in paper yml by finding the secret: field, then activating it by setting enabled: true, then finding the secret: field, then activating it by setting enabled: true, then finding the secret: field, then activating it by setting enabled: true, then finding the secret: field, then activating it by setting enabled: true, then finding the secret: field, then activating it by setting enabled: true, then finding the secret: field, then activating it by setting enabled: true, then finding the secret: field, then activating it by setting enabled: true, then finding the secret: field, then activating it by setting enabled: true, then finding the secret: field, then activating it by setting enabled: true, then finding the secret: field, then activating it by setting enabled: true, then finding the secret: field, then activating it by setting enabled: true, then finding the secret: field, then activating it by setting enabled: true, then finding the secret: field in it, and inserting enabled: true, then finding the secret: field, then activating it by setting enabled: true, then finding the secret: field, then activating it by setting enabled: true, then finding the secret: field, then activating enabled: true, then finding the secret: field, then activating enabled: true, then finding the secret: field, then activating enabled: true, then finding en
you will need to download the BungeeGuard plugin and insert the key you invented into its configured player-info-forwarding-mode = "modern" and enabled velocity support in paper.yml, this field should remain false. If you have a
bungeeguard value, it should be set to true! Next, we need to configure the servers. Everything is simple here. In the servers section you have examples of configured servers for yourself. Just specify server_name = "Server IP:server port". After you have added all the
servers you need, specify the server to which the player will be connected immediately after entering in the try = column Now let's look at some settings that you will probably change. 1) tcp-fast-open = false - set to true if you are running the server on Lunix (on a VDS/VPS/Dedicated server). 2) announce-proxy-commands = true - set it to false if you
don't want the tabcompletion to issue your proxy commands to players. 3) force-key-authentication = true - set it to false if your server is using offline mode (We don't approve of it!) Point 3 - Velocity version | shows the velocity plugins |
reloads the proxy | unloads the dump with proxy information, respectively /end - disables the proxy /shutdown - the same as end /glist - shows how many players are on proxy /server - allows you to move between servers You can read more about them here: Point 4 - Useful utilities: 1) Probably the most important part is protection from bots. This
function is performed by the following two plugins: LimboFilter + LimboApi. Actually, after installing them, everything depends on you. You can completely customize the anti-bot checks. 2) By default, there is no way to set up groups and permissions in velocity. The LuckPermsVelocity plugin will help with this. Install it on a proxy and use it as a
permission manager. 3) On velocity, as you could see, there are no commands that are in the bungeecord, such as /send, /find, etc. In order to add them, use the VelocityTools plugin. In addition to these commands, it includes useful functions, by the type of command /lobby, the ability to restrict connections directly from a digital IP, and so on. 4) The
CommandSync plugin may be quite useful for you. Recently, cycling support has been added to it, which has made it relevant again. In short, it is needed in order to execute proxy commands. If you need to
restrict other commands on the proxy server, it is recommended that you use the CommandWhitelist plugin. By the name, I think it's already clear what it does. Point 5 - Useful links. - of. velocity docks page Last edited: Apr 15, 2023 Reactions: KungfuEpt All guides on this seem outdated, on Spigot, or not explained enough for my small brain. I know
I have to add the Paper Server into my dependencies but I don't know where to go past that. I'm not quite sure how to do this, if I'll be honest. Is there any in-depth tutorial because the instructions on the GitHub have me lost... Basically, no; The test plugin repo linked is the current only example of how to set it up, documentation is planned but not a
high priority vs everything else we need to get to (pr's welcome, etc), only guide thus far is basically to clone that repo, tweak the stuff as needed, and use that repo Alright, thank you nonetheless I hate to gatekeep, but if learning from the example
github is too hard, then you will likely struggle with the compatibility issues that arise from using NMS. It's going to get a ton easier with the Mojang Mappings, but I still wouldn't recommend it unless you only plan on supporting a single version at a time, (until you understand exactly what's going on) Sorry to revive this thread again, but I've finally
got around to using the test plugin and I cloned the test plugin repo and it was going well, however, in usage, I got errors stating "Invalid plugin.yml". I put my plugin.yml, even tried letting the plugin make one itself but every time I got an error.
The only differences between this plugin and my other working one is that this one is gradle, the other is mayen, and (somehow) the project keeps forcing separate modules for my other one). @Tau your method worked, but it overrode a lot of paper methods that I have been
using, which was the main point of using the docs and other people's answers. Sorry to revive this thread again, but I've finally got around to using the test plugin and I cloned the test plugin repo and it was going well, however, in
usage, I got errors stating "Invalid plugin.yml". I put my plugin.yml in the usual spot for other plugins "src/main/resources", removed the plugin make one itself but every time I got an error. The only differences between this plugin and my other working one is that this one is gradle, the other is
maven, and (somehow) the project keeps forcing separate modules for main and test instead of using one for the entire plugin (which worked for my other one). @Tau your method worked, but it overrode a lot of paper methods that I have been using, which was the main point of using the paper, and created this weird mix between paper and spigot
that I feel would make it more difficult especially using the docs and other people's answers. Yes as i stated in my post I misread your thread. Does the jarfile contain the plugin.yml at all if you open it with a program such as 7zip? My thought is you might be relying on the maven-resources plugin to do some placeholder replacement that isn't being
done now. No, there isn't. On the test plugin it included a plugin that was supposed to "generate a plugin.yml file", but I removed it and added my own. Is there anything wrong you could point out about this image? By "no, there isn't" are you referring to the lack of a plugin.yml in the final jar? If not: If you inspect the compiled jar does the
${project.version} get replaced properly? Otherwise I wouldn't be able to help you as i'm not much of a gradle person. Yes, there is no plugin.yml. This is confusing, this setup works perfectly fine until I run it on the paperweight plugin, I'm not quite sure what could be wrong. EDIT: I just made a discovery, I decided to try and implant the plugin.yml
into the jar and it worked, but it said it could not find main class. When I looked, I realized that my entire plugin directory wasn't even there! I don't know why, I don't know how! If this is the root of the issues, then how does this even happen?! Could you please pastebin your build.gradle.kts file? And additionally what command are you running to
generate the jar? Command? I didn't know I needed a command... I've been creating artifacts and building them... or, rather: what are you clicking to build it? (screenshot is fine). Additionally, what jar are you looking at? as in where is it being output
to/what is it named. Apologies for the confusion. I created an artifact on Intellij: ^ I set the output directory straight to the plugins folder so I didn't have to move it, and the jar compiles fine (Only missing plugin.yml, but if I manually add it then it works fine) Yeah, that makes sense. By doing that you are completely bypassing paperweight and
Run. The jar will then be located in build/libs/. For distribution (running on spigot mapped servers) you'll want to select the jar without dev in the name. Thank you so much! It's working now! Reactions: Timongcraft, roan and Camm The hard fork is real! can we get much higher plays in the background Congrats to the Paper team! is tokyo still going the plays in the part to select the jar without dev in the name.
to be destroyed in the hard fork? /j Last edited: Dec 13, 2024 Reactions: Noy, Camm and kennytv This is a bad idea. now how am i going to run plugins like slimefun and geyser now? i actually read the post, i know what's going on. i know you guys want to not depend on Spigot/CraftBukkit Updates to
with our quasi closed upstream project, and have decided that we would much rather place ourselves in a position where we can work on stuff without having to worry about conflicts arising when they add features we have half a decade later. the plugins using paper-api cannot run on spigot If you use API specific to paper API, it will not work on
spigot; That has literally always been the case; long term, we will diverge, however, such is how actual forks go. i actually read the post, i know what's going on. i know you guys want to not depend on Spigot/CraftBukkit Updates to make the update faster. i still oppose this. looks terrible idea to me. if paper go independent on spigot, and if you unable
to go back to spigot, and the plugins using paper-api cannot run on spigot, it's not counted as fork anymore. it's scary. Considering 90 to 95 percent of modern servers run Paper or derivatives, it's not scary at all! It's liberating. I'm aware of dozens and dozens of plugins planning to just use paper-api, and one plugin that wants to stay on spigot. no, it's
scary to me. it's not liberating to me. what if Citizens 2 Uses New API/Methods/Enums Introduced By Spigot on their update? besides that, Citizens use Relocation. Considering 90 to 95 percent of modern servers run Paper or derivatives, it's not scary at all! It's liberating. I'm aware of dozens and dozens of plugins planning to just use paper-api, and
one plugin that wants to stay on spigot. no, it's scary to me. it's not liberating to me. what if Citizens 2 Uses New API/Methods/Enums Introduced By Spigot on their update? besides that, Citizens use Relocation. Most plugins will probably aim for working on paper considering how many servers run it. And if a dev for some reason would rather have
their plugin work on a minority of servers instead of the majority and the plugin is somewhat popular there will probably be some fork made to target paper and spigot? I am a bit inclined to oppose this decision. I don't know what content was their plugin work for both paper and spigot? I am a bit inclined to oppose this decision. I don't know what content was their plugin work for both paper and spigot? I am a bit inclined to oppose this decision. I don't know what content was their plugin work for both paper and spigot? I am a bit inclined to oppose this decision. I don't know what content was their plugin work for both paper and spigot? I am a bit inclined to oppose this decision. I don't know what content was their plugin work for both paper and spigot? I am a bit inclined to oppose this decision. I don't know what content was the plugin work for both paper and spigot? I am a bit inclined to oppose this decision. I don't know what content was the plugin work for both paper and spigot? I am a bit inclined to oppose this decision. I don't know what content was the plugin work for both paper and spigot? I am a bit inclined to oppose this decision. I don't know what content was the plugin work for both paper and spigot? I am a bit inclined to oppose this decision. I don't know what content was the plugin work for both paper and spigot? I am a bit inclined to oppose the plugin work for both paper and spigot? I am a bit inclined to oppose the plugin work for both paper and spigot? I am a bit inclined to oppose the plugin work for both paper and spigot? I am a bit inclined to oppose the plugin work for both paper and spigot? I am a bit inclined to oppose the plugin work for both paper and spigot? I am a bit inclined to oppose the plugin work for both paper and spigot? I am a bit inclined to oppose the plugin work for both paper and spigot? I am a bit inclined to oppose the plugin work for both paper and spigot? I am a bit inclined to oppose the plugin work for both paper and spigot? I am a bit inclined to oppose t
included in the bstats data that the developers reviewed when making this decision, so it seems that they may have some misunderstandings about the current situation of Paper. For many server owners, the greatest significance of using Paper is "high performance" and compatibility with Spigot, allowing them to enjoy Spigot's long-standing pluging
community while also having a modern performance server software. Even for many developers, why do I have to migrate downstream and manually narrow down the compatibility range of my plugins if there is no need for me to use the PaperAPI? That is to say, running Paper servers has a huge market share, but it does not necessarily mean that
there is a strong willingness to develop Paper plugins. In fact, I am willing to post the startup logs of my server. Let's take a look at what plugins my server (a survival oriented server that has not been modified much) consists of: [12:19:32] [ServerMain/INFO]: [PluginInitializerManager] Paper plugins (3): - BKCommonLib (1.21.4-v1-SNAPSHOT),
CrazyCrates (3.4.9), PrefiX (7.3) [12:19:32] [ServerMain/INFO]: [PluginInitializerManager] Bukkit plugins (71): - AdvancedCleaner (1.4), AnimatedScoreboard (0.3.5), AuthOp (0.1), BetterRTP (3.6.12), CMILib (1.5.2.8), Chat2QQ (1.6), CheckpointX (3.0.7), Chunky (1.4.10), Citizens (2.0.37-SNAPSHOT (build 3691)),
CommandToItem (${version}), CommonEditLib (1.2.1), CraftEnhance (2.5.6.3.3), CurveBuilding (0.6.2), DecentHolograms (2.8.12), DeluxeMenus (1.14.1-DEV-183), DisplayEntityEditor (1.0.15), Essentials (2.21.0-dev+93-3a6fdd9), Essentials (2.21.0-dev+9
ImageFrame (1.7.13.0), ItemEdit (3.5.5), ItemEdit (3.5.5), ItemEdit (3.5.5), ItemEdit (3.5.1), MochatReports (2.3.0), ODailyQuests (2.3.0-SNAPSHOT-10)
PlaceholderAPI (2.11.6), PlayerAuctions (1.28.1), PlayerParticles (8.8), ProtocolLib (5.4.0-SNAPSHOT, QuickShop-Hikari (6.2.0.7), RealisticSeasons (11.6.3), RedstoneMutex (1.0-SNAPSHOT), Residence (5.1.6.4), ScriptBlockPlus (2.3.2), Sentinel (2.9.1-SNAPSHOT), Residence (5.1.6.4), PlayerParticles (8.8), ProtocolLib (5.4.0-SNAPSHOT), Residence (5.1.6.4), ScriptBlockPlus (2.3.2), Sentinel (2.9.1-SNAPSHOT), Residence (5.1.6.4), PlayerParticles (8.8), ProtocolLib (5.4.0-SNAPSHOT), Residence (5.1.6.4), ScriptBlockPlus (2.3.2), Sentinel (2.9.1-SNAPSHOT), Residence (5.1.6.4), PlayerParticles (8.8), ProtocolLib (5.4.0-SNAPSHOT), Residence (5.1.6.4), ScriptBlockPlus (2.3.2), Sentinel (2.9.1-SNAPSHOT), Residence (5.1.6.4), ScriptBlockPlus (2.9.1-SNAPSHOT), Residence (5.1.6.4), Scri
 WeaveLumenController (1.0), WorldEdit (7.3.10-beta-01+cb9fd58), WorldEditSelectionVisualizer (2.1.6), nightcore (2.7.3), packetevents (2.7.0), pyparena (1.15.4-SNAPSHOT), qsaddon-list (6.1.0.2) TL, DR: 3 Paper plugins, 71 Bukkit (Spigot) plugins. There are many plugins here, such as Residence, Authme, DecentHolograms, DeluxeMenus and
Citizens, which are also used on many servers, but they are not the Paper version and currently do not seem interested in updating to Paper. I don't know what kind of confidence I need to have in this situation to make me believe that I should continue to follow up on the progress of Paper updates. And, This decision will undoubtedly divide the
decision, so it seems that they may have some misunderstandings about the current situation of Paper. For many server owners, the greatest significance of using Paper is "high performance" and compatibility with Spigot, allowing them to enjoy Spigot's long-standing plugin community while also having a modern performance server software. Even
for many developers, why do I have to migrate downstream and manually narrow down the compatibility range of my plugins if there is no need for me to use the Paper API? That is to say, running Paper servers has a huge market share, but it does not necessarily mean that there is a strong willingness to develop Paper plugins. In fact, I am willing to
post the startup logs of my server. Let's take a look at what plugins my server (a survival oriented server that has not been modified much) consists of: [12:19:32] [ServerMain/INFO]: [PluginInitializerManager] Paper plugins (3): - BKCommonLib (1.21.4-v1-SNAPSHOT), CrazyCrates (3.4.9), Prefix (7.3) [12:19:32] [ServerMain/INFO]:
[PluginInitializerManager] Bukkit plugins (71): - AdvancedCleaner (1.4), AnimatedScoreboard (0.3.5), AuthMe (5.6.0-beta2-b2453), AuthOp (0.1), BetterRTP (3.6.12), CMILib (1.5.2.8), Chat2QQ (1.6), CheckpointX (3.0.7), Chunky (1.4.10), Citizens (2.0.37-SNAPSHOT (build 3691)), CommandToItem (${version}}), CommonEditLib (1.2.1), CraftEnhance
(2.5.6.3.3), CurveBuilding (0.6.2), DecentHolograms (2.8.12), DeluxeMenus (1.14.1-DEV-183), DisplayEntityEditor (1.0.15), Essentials (2.21.0-dev+93-3a6fdd9), ExcellentEnchants (4.3.3), ExtraContexts (2.0-SNAPSHOT), GSit (1.12.1), ImageFrame (1.7.13.0), ItemEdit (3.5.5), ItemTag (3.5.1), Link2QQ (1.2),
LuckPerms (5.4.131), MechanicsCore (4.0.1), MiniMOTD (2.1.0), MiraiMC (1.8-rc1), MoarBows (2.6), MobArena (0.108), MobsToEggs (1.7.14), Multiverse-Core (4.3.13), Multiverse-Portals (4.2.3), NexEngine (2.2.12), NoChatReports (2.3.0-SNAPSHOT-10), PlayerAuctions (1.28.1), PlayerParticles (8.8)
ProtocolLib (5.4.0-SNAPSHOT, 739), Quests (3.15.2-b216e2b), QuickShop-Hikari (6.2.0.7), RealisticSeasons (11.6.3), RedstoneMutex (1.0-SNAPSHOT (build 523)), SetSpawn (4.8), SkinsRestorer (15.5.1), SpeedRoads (1.0.1), StartupCommands (0.0.6), TAB (5.0.3), RedstoneMutex (1.0-SNAPSHOT), Residence (5.1.6.4), ScriptBlockPlus (2.3.2), Sentinel (2.9.1-SNAPSHOT), Residence (5.1.6.4), ScriptBlockPlus (
WorldEditSelectionVisualizer (2.1.6), nightcore (2.7.3), packetevents (2.7.0), pyparena (1.15.4-SNAPSHOT), qsaddon-list (6.1.0.2) TL, DR: 3 Paper plugins, 71 Bukkit (Spigot) plugins. There are many plugins here, such as Residence, Authme, DecentHolograms, DeluxeMenus and Citizens, which are also used on many servers, but they are not the
Paper version and currently do not seem interested in updating to Paper. I don't know what kind of confidence I need to have in this situation to make me believe that I should continue to follow up on the progress of Paper updates. And, This decision will undoubtedly divide the communities of Spigot and Paper in the long run, forcing all plugin
developers to face a "choice between two" situation. As far as the current situation is concerned, I cannot consider this a wise decision. agreed. i oppose this. we hate to face choice between 2. see? it's scary. "Paper plugin" is sort of bad branding on our side, it does not mean that any of the other plugins are not using Paper API, it's simply a separate
(optional!) plugin loader with more features. Most of your plugins already use some form of Paper API or even strongly recommend using Paper. why do I have to migrate downstream and manually narrow down the compatibility range of my plugins if there is no need for me to use the PaperAPI? Hence this is a bit of a misnomer; not using Paper.
already limits a lot of your plugins' functionality. I can guarantee you that both devs and Paper users won't get the short end of the stick once the two platforms diverge further. I understand that such change can be scary, but you really have nothing to worry about, and there's hardly a difficult choice to be made here. Either you add simple
compatibility measures if you really need to support those remaining few %, or you simply swap a dependency only for new versions going forward Afaik paper plugins are also currently in the process of being rebranded to Lifecycle plugins which is a more accurate term. Your plugin does not need to be a paper/Lifecycle plugin to work on paper
infact we recommend people do not use this plugin loader unless you need the Features it provides For many server owners, the greatest significance of using Paper is "high performance" and compatibility with Spigot, allowing them to enjoy Spigot's long-standing plugin community while also having a modern performance server software. Even for
many developers, why do I have to migrate downstream and manually narrow down the compatibility range of my plugins if there is no need for me to use the PaperAPI? We're not going to break existing APIs for some sizable period of time; the only immediate difference is us not merging new APIs, but most of the side effects of this have been
navigated by plugin developers for years, given that they're often expected or desire to support multiple versions of the software. We trust that the developers who are willing are smart enough to figure out how they'd want to support two platforms should they desire. This is especially true when plugins are increasingly having to rely on fragile
operations to take advantage of Mojang's infra improvements in the server. That is to say, running Paper servers has a huge market share, but it does not necessarily mean that there is a strong willingness to develop Paper plugins. In fact, I am willing to post the startup logs of my server. Let's take a look at what plugins my server (a survival oriented
server that has not been modified much) consists of: [12:19:32] [ServerMain/INFO]: [PluginInitializerManager] Paper plugins (3): - BKCommonLib (1.21.4-v1-SNAPSHOT), CrazyCrates (3.4.9), Prefix (7.3) [12:19:32] [ServerMain/INFO]: [PluginInitializerManager] Bukkit plugins (71): - AdvancedCleaner (1.4), AnimatedScoreboard (0.3.5), AuthMe
(5.6.0-beta2-b2453), AuthOp (0.1), BetterRTP (3.6.12), CMILib (1.5.2.8), Chat2QQ (1.6), CheckpointX (3.0.7), Chunky (1.4.1-DEV-183), CraftEnhance (2.5.6.3.3), CurveBuilding (0.6.2), DecentHolograms (2.8.12), DeluxeMenus (1.14.1-DEV-183),
DisplayEntityEditor (1.0.15), Essentials (2.21.0-dev+93-3a6fdd9), 
(2.6), MobArena (0.108), MobSToEggs (1.7.14), Multiverse-Core (4.3.13), Multiverse-Core (4.3.13), Multiverse-Core (4.3.13), Multiverse-Core (4.3.13), PlayerParticles (8.8), ProtocolLib (5.4.0-SNAPSHOT-739), Quests (3.15.2-b216e2b), QuickShop-Hikari (6.2.0.7), PlayerParticles (8.8), ProtocolLib (6.2.0.7), PlayerParticles (8.8), PlayerParticle
RealisticSeasons (11.6.3), RedstoneMutex (1.0-SNAPSHOT), Residence (5.1.6.4), ScriptBlockPlus (2.3.2), Sentinel (2.9.1-SNAPSHOT), Residence (1.0.1), StartupCommands (0.0.6), TAB (5.0.3), TerraformGenerator (18.0.0), TianyuQQBinder (1.0), Train_Carts (1.21.4-v1-SNAPSHOT), Residence (5.1.6.4), ScriptBlockPlus (2.3.2), Sentinel (2.9.1-SNAPSHOT), Residence (5.1.6.4), ScriptBlockPlus (2.9.1-SNAPSHOT), Residence (5.1.6.4), Residence (5.1.6.4), Residence (5.1.6.4), Residence (5.1.6.4), Residenc
Traincarts2Dynmap (1.0), Vault (1.7.3-b131), VeinMiner (2.2.6), ViaBackwards (5.2.2-SNAPSHOT), ViaVersion (5.2.2-SNAPSHOT), WeaveCustomSchedule (1.0), WorldEdit (7.3.10-beta-01+cb9fd58), WorldEditSelectionVisualizer (2.1.6), nightcore (2.7.3), packetevents (2.7.0), pvparena (1.15.4-beta-01+cb9fd58), which is a second of the control of
SNAPSHOT), qsaddon-list (6.1.0.2) TL, DR: 3 Paper plugins, 71 Bukkit (Spigot) plugins. This is a misnomer; paper plugins are more just "advanced bootstrap earlier and perform various operations, such as injecting entries into the servers' registry, something which is going to become
increasingly important over the next (and past) few years as mojang is data driving the server software. Paper already has native support for plugins to requirement for plugins to adapt to this system; in fact, we actively discourage people
from using this system unless they need to use it for something. The reality is that most plugins do not need to opt into using the early bootstrapper stuff, but it's there for the plugins that want to. I don't know what kind of confidence I need to have in this situation to make me believe that I should continue to follow up on the progress of Paper
updates. And, This decision will undoubtedly divide the communities of Spigot and Paper in the long run, forcing all plugin developers to face a "choice between two" situation. As far as the current situation is concerned, I cannot consider this a wise decision. Spigot hasn't cared to interact with the community outside of its little cubby hole for well
over a decade now, and the last noticeable interaction we had with the project was an "if you start sending us all your patches and kill off Paper, I might let you join the spigot team". The only real community overlap I've seen is active paper developers going over to Spigot to try to collaborate on aspects like API design that impact us, such as the
Item/BlockType APIs, which are much more in line with our original vision we had 10 years ago. The only reach-out we ever got from there these days is the odd contributor who wants our input on something they worked on. -- We have so many patches over Spigot that we weren't able to explore our patches folder on GitHub for the past year, and the
paralysis of "but, spigot might expose API for that" often made us stall working on or merging PRs for such features, such as the well-needed separation from that broken API concept), something Spigot finally accepted needed to happen,
almost 10 years after core project leaders in Paper identified the need for this. Mojang's recent changes are also "fun" in that releases are more frequent and much heavier on the workload in order to rewrite parts of the game, especially when dealing with the unstable output provided by upstreams' tooling which frequently caused conflicts due to
them using a much older decompiler which produced more unstable and less 'unrefined' output, we have a fraction of the decompile that spigot performs. Thanks to taking advantage of and collaborating with projects within the community, our source tree is
much nicer to work with. The nature of Mojang's new release cycle also allows them to release new features and iterate on them. Rather than the ping-pong, pong we'd get where you'd deal with a substantially large update followed up by what are often smaller fixes, the releases are now more ping-pong, ping-pong. This means that we're
no longer dealing with one or two substantial updates a year combined with a few quick ones but multiple sizable releases a year. Part of the reason we've been able to manage this thus far is more a set of happenchances; we can't start to work on anything until our quasi-closed-source upstream performs a release, by luck, the timezones generally
mean that this is an early afternoon for us Europeans, assuming he doesn't hold a release hostage, we can generally preplan to some degree for people to be around to start, however, it's only through the fact that our team we've managed to build is crazily committed to developing this software for the community and is willing to pull off the crazy
hours and work around the clock, which is honestly so unsustainable I'm surprised that we've managed to pull it off for as long as we have. -- It's also worth remembering that one of the first changes Mojang did on this release cycle was rewriting the entirety of how ItemStacks are represented, migrating from "just a blob of NBT" into a much more
structured and defined manner of representing this data, with items now defining their data using prototypes rather than sporadically hardcoded behaviour across the codebase; These changes were probably one of the more notable divergences from Spigot we've had here; 1) The ItemMeta API that they're so insistent on refusing to migrate away
from is broken and fails to represent how items work in the game, i.e. you cannot query attributes about a food item in vanilla in their API; only items with that component overridden by plugins, manually; That means that you can't query components which are actually on the Item itself and would thus need to use internals or hardcoding, to perform
such operations. 2) Spigot opted for compatibility of legacy API behaviour in a manner that transforms "I want to disable all of the default attributes on this item, turning all of my tools into performing worse than wooden tier tools", rather than actually addressing the fact that ItemStacks are a bit less
malleable on this front. In terms of working with the community rather than sticking in our corner, we've been able to highlight to Mojang that many members and we dislike this underlying change as it creates a bunch of issues; as for what Mojang will do with that, we're yet to see, but, at least we know they know the discontent of the bundling of
some of this data on their components. It's worth noting that we already saw the signs on the wall and had started work on a Data Component API when Mojang released this change, a welcomed happy surprise that we were able to adopt directly to cater towards that underlying data representation relatively easily; And that's before you get into API
that Spigot generally refused to support for the better part of a decade, i.e. the Biome API only just got support for custom biomes, something we've had primitive support for the better part of a decade, i.e. the Biome API only just got support for use of a decade, i.e. the Biome API only just got support for custom biomes, something we've had primitive support for in Paper for years, but even as much as we wanted to, couldn't work on replacing it as, at first, we'd have to break API compatibility; Then, after a few years, they finally
accepted that it needed to happen, in which, we where then stuck on waiting for them to decide if and how they're going to do it; Do they go the route that allows us to deprecate the older API and move forward with the new, and properly expose the underlying data represented by Biomes, or, do they go the route that keeps legacy plugins work for a
millennium and fail to expose any modern information about Biomes to plugins. They, of course, opted for the latter. -- As much as I'd love to say, "Paper is forever," one day, I and the many others I interact with on this project daily will one day get too busy and have to move on. For that, we will need to be able to onboard new developers and find
people who are able and willing to replace myself and other team members as the time arises. There are many people out there who fit that bill, but the community and other developers for years. Mojang's new release cycles, our
desire to work on supporting many of the new things Mojang is increasingly offering in the game, and dealing with the fact that we were struggling to onboard new developers means that we hit our "now or never"; and now, rather than worrying about how we might have to wind this project down over the next 5 years, We're more focused on
ensuring that the next 5 years is spent on preparing for the next 5 years after that. The hard fork was pretty essential to the life of this project, and we hope that many are willing to join the ride with us over the next many years, but, it is ultimately your server. -- NB: apologies that this got kinda rambly Reactions: Going and LoJoSho We're not going
to break existing APIs for some sizable period of time; the only immediate difference is us not merging new APIs, but most of the side effects of this have been navigated by plugin developers who are willing are smart enough
to figure out how they'd want to support two platforms should they desire. This is especially true when plugins are increasingly having to rely on fragile operations to take advantage of Mojang's infra improvements in the server. This is a misnomer; paper plugins are more just "advanced bootstrapper lifecycle plugins", offering access for plugins to
bootstrap earlier and perform various operations, such as injecting entries into the servers' registry, something which is going to become increasingly important over the next (and past) few years as mojang is data driving the server software. Paper already has native support for plugins to register custom enchantments, painting variants, damage
types, and more over the next few years. There is no requirement for plugins to adapt to this system; in fact, we actively discourage people from using the early bootstrapper stuff, but it's there for the plugins that want to. Spigot hasn't
cared to interact with the community outside of its little cubby hole for well over a decade now, and the last noticeable interaction we had with the project was an "if you start sending us all your patches and kill off Paper, I might let you join the spigot toan to Spigot to
try to collaborate on aspects like API design that impact us, such as the Item/BlockType APIs, which are much more in line with our original vision we had 10 years ago. The only reach-out we ever got from there these days is the odd contributor who wants our input on something they worked on. -- We have so many patches over Spigot that we
weren't able to explore our patches folder on GitHub for the past year, and the paralysis of "but, spigot might expose API for that" often made us stall working on or merging PRs for such features, such as the well-needed separation from
that broken API concept), something Spigot finally accepted needed to happen, almost 10 years after core project leaders in Paper identified the need for this. Mojang's recent changes are also "fun" in that releases are more frequent and much heavier on the workload in order to rewrite parts of the game, especially when dealing with the unstable
output provided by upstreams' tooling which frequently caused conflicts due to them using a much older decompile fixes that spigot needs with the entire source subtree decompiled vs the partial decompile that spigot performs. Thanks to taking
advantage of and collaborating with projects within the community, our source tree is much nicer to work with. The nature of Mojang's new release cycle also allows them to release new features and iterate on them. Rather than the ping-pong, pong we'd get where you'd deal with a substantially large update followed up by what are often
smaller fixes, the releases are now more ping-pong, ping-pong, ping-pong. This means that we're no longer dealing with one or two substantial updates a year combined with a few quick ones but multiple sizable releases a year. Part of the reason we've been able to manage this thus far is more a set of happenchances; we can't start to work on anything until our
quasi-closed-source upstream performs a release, by luck, the timezones generally mean that this is an early afternoon for us Europeans, assuming he doesn't hold a release hostage, we can generally preplan to some degree for people to be around to start, however, it's only through the fact that our team we've managed to build is crazily committed
to developing this software for the community and is willing to pull off the crazy hours and work around the clock, which is honestly so unsustainable I'm surprised that we've managed to pull it off for as long as we have. -- It's also worth remembering that one of the first changes Mojang did on this release cycle was rewriting the entirety of how
 ItemStacks are represented, migrating from "just a blob of NBT" into a much more structured and defined manner of representing this data, with items now defining their data using prototypes rather than sporadically hardcoded behaviour across the codebase; These changes were probably one of the more notable divergences from Spigot we've had
here; 1) The ItemMeta API that they're so insistent on refusing to migrate away from is broken and fails to represent how items with that component overridden by plugins, manually; That means that you can't query components which are actually on
the Item itself and would thus need to use internals or hardcoding, to perform such operations. 2) Spigot opted for compatibility of legacy API behaviour in a manner that transforms "I want to disable all of the default attributes on this item, turning all of my tools into performing worse than wooden tier
tools", rather than actually addressing the fact that ItemStacks are a bit less malleable on this front. In terms of working with the community rather than sticking in our corner, we've been able to highlight to Mojang that many members and we dislike this underlying change as it creates a bunch of issues; as for what Mojang will do with that, we're
yet to see, but, at least we know they know the discontent of the bundling of some of this data on their components. It's worth noting that we already saw the signs on the wall and had started work on a Data Component API when Mojang released this change, a welcomed happy surprise that we were able to adopt directly to cater towards that
underlying data representation relatively easily; And that's before you get into API that Spigot generally refused to support for the better part of a decade, i.e. the Biome API only just got support for uson biomes, something we've had primitive support for in Paper for years, but even as much as we wanted to, couldn't work on replacing it as, at first,
we'd have to break API compatibility; Then, after a few years, they finally accepted that it needed to happen, in which, we where then stuck on waiting for them to decide if and how they're going to do it; Do they go the route that allows us to deprecate the older API and move forward with the new, and properly expose the underlying data represented
by Biomes, or, do they go the route that keeps legacy plugins work for a millennium and fail to expose any modern information about Biomes to plugins. They, of course, opted for the latter. -- As much as I'd love to say, "Paper is forever," one day, I and the many others I interact with on this project daily will one day get too busy and have to move on.
For that, we will need to be able to onboard new developers and find people who are able and willing to replace myself and other team members as the time arises. There are many people out there who fit that bill, but the project's stalemate was just not conducive to that. The hard fork isn't something new; it has been discussed with the community
and other developers for years. Mojang's new release cycles, our desire to work on supporting many of the new things Mojang is increasingly offering in the game, and dealing with the fact that we were struggling to onboard new developers means that we hit our "now or never"; and now, rather than worrying about how we might have to wind this
project down over the next 5 years, We're more focused on ensuring that the next 5 years is spent on preparing for the next 5 years after that. The hard fork was pretty essential to the life of this project, and we hope that many are willing to join the ride with us over the next 5 years, but, it is ultimately your server. -- NB: apologies that this got
kinda rambly I'm glad to hear these interesting news about Paper development. As a developer, I fully understand the difficulty of Paper's work in this situation, as well as the need to strike a difficult balance between "supporting new features" and "annoying upstream code" every time. I must also admit that Paper provides more modern and
numerous features than Spigot, and to be honest, many of these Paper features should have been the work that Spigot needs to do. Indeed, many of those Bukkit plugins also use Paper API to complete their functions - after all, not all plugin developers are always willing to deal with NMS. I admit that my concerns are somewhat unnecessary for
excellent Paper developers, but I think the content mentioned in the community to enhance people's confidence (or developer interviews or the like). It's not a bad thing to talk about such things in a rambling manner, as there is too much
background information that needs to be explained to the audience. But obviously, currently Paper development still relies on some packages from org.bukkit. Perhaps one day you will replace them all with io.papermc.paper: ) Hey, it seems that no matter what, we need to say goodbye to Spigot. But i still dont like that! It's still scary... I use the
Spigot API Based Plugins more often. for that, i would like to still stay in the spigot-api Plugins. I dont want to use Geyser-Standalone. Slimefun will be
Using Paper-API is so low. So i am still scared about it. But i still don't like that! It's still scary... I use the Spigot API Based Plugins. I don't want to use alternatives of my favorite plugins. Geyser-Spigot also uses Spigot For Some reason, i don't want to use Geyser-Standalone. Slimefun
has Paper Support Through PaperLib, But it has Spigot Support. The chances that Slimefun will be Using Paper-API is so low. So i am still scared about it. It really feels like there's no scenario that would make you happy other than promising 100% spigot compatibility forever. If some well-liked plugins choose to support only a tiny fraction of their
users, I'm sure alternatives (or, even, just forks) will appear to have paper support. I'm a pretty anxious person who spends far too much of his time thinking about negative what-ifs and running thought experiments in my head about how things can go wrong. I'm not worried about paper's hardfork. If some well-liked plugins choose to support only a
tiny fraction of their users, I'm sure alternatives (or, even, just forks) will appear to have paper support. alternatives designed for paper dont work like original ones. Slimefun alternatives (or, even, just forks) will appear to have paper support. alternatives designed for paper dont work like original ones. Slimefun alternatives (or, even, just forks) will appear to have paper support.
designed for paper dont work like original ones. Slimefun alternatives Such as itemedit doesn't have Industry related things. I did also say fork in the message you quoted. That's also a possibility, for if a plugin chooses to only support a couple percent of their users, abandoning 95% of them, i am super scared about paper's future, what if
bukkit/spigot compatblity breaks? I do not know what to tell you, as you appear unwilling to accept any possibility but catastrophe. of course i am unwilling to accept any possibility but catastrophe. there is no better alternatives for slimefun. No One will try to rewrite Slimefun IV and some of the alternatives use commands or configs(which are
vulnerable) instead of Java Code. of course i'm scared of it. sorry for off-topic by the way. I don't know why your personal choice of software to use becomes a responsibility of Paper itself. Either embrace it or be left in forgotten history. The world moves on
without you or me - that's just how things are. maybe i will consider embracing it... i have no choice, but to say goodbye to our Spigot support in favor of Paper, so I honestly am not sure where all the fear comes from. Did you perhaps
missed their announcement? oh, thank god it's dropping support for spigot now. ok, now i understand. i have to accept the hardfork. and maybe, adapt to new modern API. i wasn't aware about the announcement because i didnt have a discord account. Reactions: EterNity is there a chance that Spigot API get replaced with new api and Dropped in the
future when Paper Plugin API is complete? No API that isn't already marked as deprecated for removal will be removed well, seems that no matter what, we must Move Away from SpigotMC Forum, Reobfuscated Mapping Server, to Paper-API,
PaperMC Forum, hangar&Modrinth, Mojang Mapping, This is an Exodus from Spigot API, Spigot Forum, The Destination is PaperMC, This Is An Exodus From Spigot API, Spigot Forum, The Destination is PaperMC, This Is An Exodus From Spigot API, Spigot Forum, The Destination is PaperMC Forum, The Destination is PaperMC, This Is An Exodus From Spigot API, Spigot Forum, The Destination is PaperMC Forum, The Destination is PaperMC Forum, The Destination is PaperMC Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exodus From Spigot API, Spigot Forum, This is an Exo
Feb 21, 2025 Hello I've been following this closely and have read the entire thread, including the responses from the Paper developers. I understand the purpose of this hardfork, but I agree with some other admins who are concerned because my server relies on numerous plugins to deliver its content. So I submitted the question to a few developers.
to anticipate the impact this will have on my server and to see which plugins won't follow Paper when it decouples from Spigot. Furthermore, supporting only Paper would prohibit them from using Spigot site, which remains the most
popular and well-known site for distributing their plugins. Despite the fact that Paper is now much more widely used than Spigot (63% vs. 16%), they prefer to continue supporting Spigot... In short, all this is to say that our concerns remain valid at this time, and if the Paper hardfork breaks compatibility with the Spigot API in future, it's undeniable
that many servers will find it very difficult to continue updating their plugins. Especially those that use a lot of them, including some that don't have an equivalent. For us, the question to ask is whether it's more viable for our servers to switch back to Spigot while there's still time, or whether we take the risk of following Paper. It's a difficult gamble,
fraught with uncertainty: not to mention the servers that have no choice but to remain on Paper due to its high resource consumption. Anyway, thanks for your great work on Paper we understand your concerns that some of the plugins you're currently using might no longer work on Paper in the future because the developers will only support Spigot.
However, we know of many plugins that will either keep supporting both Spigot and Paper or outright only support Paper in the future, such as WorldGuard, EssentialsX, and Geyser. Furthermore, for modern versions, there is an even larger divide between Spigot and Paper. For example, for 1.21.4, the amount of servers using Paper is
>95%, so we're confident that many plugins will keep supporting Paper. Of course there might be some developers that will only support Spigot, but we expect the community to be able to provide forks or alternatives for these plugins for Paper. Last edited: Mar 30, 2025 A bigger concern I think is not plugins that are still being supported, but those
that aren't. Like if something in the paper api breaks vault for example even though that's unlikely. Although that's unlikely normal Minecraft updates tend to break plugins that rely on NMS anyway if they aren't updated. When it comes to places where plugins can be downloaded, I doubt Hanger is going to become that popular at least not for a while. I
find that more and more new plugins are being posted on Modrinth, but there are other alternatives like Built by Bit. The only reason why the spigot site is popular is because it was created years before Paper. Due to the cross compatability that new plugins strived for, they decided just to use the same website everyone else was using. That being
said, Hanger needs some SEO work to become popular. Even the Paper plugins on my server I don't download from Modrinth. Modrinth has a huge SEO advantage mostly due to it being a large platform already for modding. Maybe there is something Paper could offer plugin devs to get them to switch over
to Hanger instead of Modrinth? There is already project importing from Spigot but it could be advertised more. Either way, Modrinth is a good platform for paper plugins already and I bet most plugins that now only want to support paper will just switch over to Modrinth if not to multiple sites. I'm really glad you finally decided to do this! I don't
understand the people who are against separating from Spigot — Spigot isn't going anywhere, and if someone wants to stay on it, they still can. At the same time, for those who truly want to grow and move forward with new possibilities, this is a great decision! Ideally, I'd love to see the Paper API become completely different from the Spigot/Bukkit
API — to maximize usability, performance, and embrace a modern approach. I'd like the difference between Spigot and Paper to be as big as the difference between BungeeCord and Velocity. Thank you for your work, and good luck to you guys! Hello, Only the owners of small or new servers can rejoice in such an unreasonable decision. Probably also
those who can afford to pay developers a fortune to update their plugins, sometimes abandoned but still working, will find themselves trapped in a fratricidal war between the developers of Paper and Spigot. Basically to show who has the biggest one.. When Paper
was released, the deal was that all Spigot plugins would remain compatible, otherwise they knew the admins would stay on Spigot. We, the admins, trusted them, and now that they've tricked us, they want to destroy this very important compatibility. Most of those reading this won't understand because they haven't been involved in Minecraft servers
for 15 years, and won't be able to grasp the issues involved. A well-thought-out and reasonable solution would instead be to create a new fork of Paper (ModernPaper, for example) while maintaining the current Paper branch compatible version of
Paper because without it, servers like mine will simply disappear because the majority of their plugins exclusive to Paper is ridiculous compared to the Spigot-compatible offering. Hangar is also extremely poorly designed, while the Spigot website is a benchmark in its field; everything is clear, each
plugin has its own discussion thread, including comments, whereas on Hangar, everything is decentralized; we don't feel like we're part of a community, unlike on the Spigot website. Plus ther's no activity on forums. There's no support on the website, but it redirects to the Discord pages for each plugin, except that Discord doesn't allow you to have
more than 100 servers (except for those who can afford to waste their money on Nitro...). Now that I've trusted Paper in the past, I'm stuck; my server will no longer run on Spigot because I've benefited from Paper's performance while having confidence in it. So I can't even go back to Spigot if I want to keep my 14-year-old server alive, and even
though no one here cares, I'm disgusted because it's going to destroy 14 years of work. Why? Because developers are bored and want to turn a 1950s car into a post-2000s prototype. All we want is for our servers to actually work. If we want to turn a 1950s car into a post-2000s prototype.
my concerns. Contrary to what you might think, I'm not angry—I'm a nice girl—but I'm just disgusted and very worried that my community's adventure is coming to an end because of an unreasonable decision Have a good Sunday Reasonable server owners always keep their servers updated. Paper has always supported only the latest version and
nothing else. Because of that, I don't see how any of your points are valid. Older versions aren't supported in the first place, hardfork or not. There are plenty of plugins available on both Hangar and Modrinth these days. Spigot is no longer the only place for resources. If you didn't know that, you probably haven't been keeping up with the latest
developments in the Minecraft space. Running an outdated server is like using Windows XP in 2025. It's insecure, full of exploits, and prone to crashes, since only the latest version receives patches. I feel like you are the one being unreasonable here. Project moves on and being forever stuck supporting legacy will only stagnant the development
which is exactly why Spigot has been left behind in first place (as you mentioned yourself in the post, you cannot live without Paper's performance patch). I'd just like to point out that, much like we may someday do when enough time has passed, spigot has regularly broken older spigot has been left behind in first place (as you mentioned yourself in the post, you cannot live without Paper's performance patch). I'd just like to point out that, much like we may someday do when enough time has passed, spigot has been left behind in first place (as you mentioned yourself in the post, you cannot live without Paper's performance patch). I'd just like to point out that, much like we may someday do when enough time has passed, spigot has been left behind in first place (as you mentioned yourself in the post, you cannot live without Paper's performance patch). I'd just like to point out that, much like we may someday do when enough time has passed, spigot has been left behind in first place (as you mentioned yourself in the post, you cannot live without Paper's performance patch). I'd just like to point out that, much like we may someday do when enough time has passed in the post, you cannot live without Paper's performance patch in the post, you cannot live without Paper's performance patch in the post, you cannot live without Paper's performance patch in the post, you cannot live without Paper's performance patch in the post, you cannot live without Paper's performance patch in the post, you cannot live without Paper's performance patch in the post, you cannot live without Paper's performance patch in the post, you cannot live without Paper's performance patch in the post, you cannot live without Paper's performance patch in the post, you cannot live without Paper's performance patch in the post, you cannot live without Paper's performance patch in the post, you cannot live without Paper's performance patch in the post, you cannot live without Paper's performance patch in the post, you cannot live without Paper's per
and ancient plugins have always eventually broken. Vault touches one tiny piece of API that has remained untouched and is an exception. You are definitely taking a fairly small impact on servers and exaggerating it to a scary event. Please do not be afraid. Things will be okay and we are here to support you through the gentle changes. Visit us on
```

```
discord if you want more of a conversation than the slowness of forum posts. My server is updated to latest game version. But not all of Reasonable server owners always keep their servers updated. Paper has always supported only the latest version and nothing else. Because of that, I don't see how any of your points are valid. Older versions aren't
supported in the first place, hardfork or not. There are plenty of plugins available on both Hangar and Modrinth these days. Spigot is no longer the only place for resources. If you didn't know that, you probably haven't been keeping up with the latest developments in the Minecraft space. Running an outdated server is like using Windows XP in 2025.
It's insecure, full of exploits, and prone to crashes, since only the latest version receives patches. I feel like you are the one being unreasonable here. Project moves on and being forever stuck supporting legacy will only stagnant the development which is exactly why Spigot has been left behind in first place (as you mentioned yourself in the post, you
cannot live without Paper's performance patch). I think there's a misunderstanding^ If the future of Paper worries me, it's precisely because I keep my server up to date. Otherwise, I wouldn't worry about it! My server has always been kept up to date since its creation and is currently at 1.21.4. The map hasn't been reset for 14 years, yet I keep it up
to date, without deleting a single build. I work like crazy to keep my server up to date like you've never seen anywhere else, I assure you. So no, the problem is the many plugins we use, many of which haven't been
maintained for years but still work. My server has always been free for its users (no pay-to-win, no store, nothing!), but it costs me a fortune, and my small salary won't allow me to pay for so many updates. And I'm not the only one in this situation. Of course, I've been aware of the various sites for downloading plugins for a long time, and I even saw
their creation, just like Paper, but that doesn't change anything. There's no equivalent for many of my plugins, and the offering is much smaller and less extensive than Spigot. Even if more anything. There's no equivalent for many of my plugins, and the offer is spread over
several sites instead of being on the same single site such as Spigot. The other problem is that we use many niche plugins, which are part of the specific features of my server and the developers a bit, and this hard fork will completely divide the community. A
large number of developers won't follow through, even though their plugins are paid. Others don't really know what they'll do, and others will simply stop developing plugins. I'm not the one saying this stupidly, but rather them when I contacted them about my concerns. Paper forgets where it comes from, its origins, and what made its existence
possible. I'd just like to point out that, much like we may someday do when enough time has passed, spigot has regularly broken older spigot plugins. This isn't new. Devs have always had to update things, and ancient plugins have always eventually broken older spigot plugins. This isn't new. Devs have always had to update things, and ancient plugins have always eventually broken. You are
definitely taking a fairly small impact on servers and exaggerating it to a scary event. Please do not be afraid. Things will be okay and we are here to support you through the gentle changes. Visit us on discord if you want more of a conversation than the slowness of forum posts. It's true that I'm worried because my server is the foundation of a very
nice, close-knit community, and if I can't update my server because most of its unique features will no longer be compatible. I've worked so hard to maintain this community for almost 15 years that, of course, such
an announcement worries me more than anything else. I'll probably join your Discord channel. Maybe it'll do me some good to chat with you. Sorry if I came across as rude; that's not my intention, I'm just scared. Older servers that rely on numerous plugins, sometimes abandoned but still working, will find themselves trapped in a fratricidal war
between the developers of Paper and Spigot. A lot of these plugins only still work due to Spigots attitude towards never breaking plugins ever (except for the random coin toss in which they broke API in manners that could cause duplication issues or just crash servers with 0 warning mid-release cycle), to the degree that they invested the effort into a contract of the could cause duplication issues or just crash servers with 0 warning mid-release cycle), to the degree that they invested the effort into a contract of the could cause duplication issues or just crash servers with 0 warning mid-release cycle), to the degree that they invested the effort into a contract of the could cause duplication issues or just crash servers with 0 warning mid-release cycle), to the degree that they invested the effort into a contract of the could cause duplication issues or just crash servers with 0 warning mid-release cycle).
chonky mechanism which rewrites the code of every single plugins that loads, ever; and have often refused to expose features to plugins over being able to do basic operations like query the biome of a chunk. They finally accepted that this would harm their ability to move forward and eventually began working on
some of the things the developer community had been asking them to do for a decade. The bytecode rewriting stuff is sadly something the ecosystem forward. When Paper was released, the deal was that all Spigot plugins would remain
compatible, otherwise they knew the admins would stay on Spigot. We, the admins, trusted them, and now that they've tricked us, they want to destroy this very important compatibility. At the time, that was a goal we considered beneficial for both the community and ourselves. In good faith, we cannot refuse to move forward with the Minecraft
ecosystem. We either adopt Mojang's efforts to improve the power available to server owners or give up and close shop. We have consistently talked about our plans to break away from spigot for years now, in public, there is no tricking here, we just finally decided we could no longer remain in our current situation and decided that it was time to
deliver on that promise. A well-thought-out and reasonable solution would instead be to create a new fork of Paper (ModernPaper, for example) while maintaining the current Paper branch compatible with the Spigot API. Reasonable? Are you paying us for this extra effort that would double, if not triple, the amount of work we're doing? A large part of
the reason why we hard forked was because the maintanance over the top of spigot was becoming increasingly unbearable; The current update has some things that we need to look into writing API for stuff that's needed going forward. This was
only going to increasingly become an issue as mojang starts working more on various things, such as making it possible for servers to define custom Items, and as they work on their data component system, something which spigot seemingly has 0 intent of actually supporting properly. We're only planning to expose more features towards the API in
order to allow them to do things that require API breaks in order to pull off, such as, when mojang delivers on the framework, the support for custom items, something which is literally impossible to handle using the currently supported API from upstream. Hangar is also extremely poorly designed, while the Spigot website is a benchmark in its field;
everything is clear, each plugin has its own discussion thread, including comments, whereas on Hangar is generally a work in progress, and many other platforms are out there, as noted. There is also places for feedback on suggestions and
improvements, though, some aspects, like commands, are a bit of a concern more in terms of policy (i.e. do we try to manage aspects of moderation that spigot refuses to deal with). I do not know a single developer who considers Spigot's
resource section to be a benchmark. If anything, during my years of being there, I saw consistent complaints and feature requests against that system, which Spigot is unable to fulfil. it's a Forum running a plugin that allows people to list stuff and doesn't support any basic feature that exists within the rest of the entire ecosystem that allows people to list stuff and doesn't support any basic feature that exists within the rest of the entire ecosystem that allows people to list stuff and doesn't support any basic feature that exists within the rest of the entire ecosystem that allows people to list stuff and doesn't support any basic feature that exists within the rest of the entire ecosystem that allows people to list stuff and doesn't support any basic feature that exists within the rest of the entire ecosystem that allows people to list stuff and doesn't support any basic feature that exists within the rest of the entire ecosystem that allows people to list stuff and doesn't support any basic feature that exists within the rest of the entire ecosystem that allows people to list stuff and doesn't support any basic feature that exists within the rest of the entire ecosystem that allows people to list stuff and doesn't support any basic feature that exists within the rest of the entire ecosystem that exists a support and the exist suppor
begging for for years, such as the ability to have teams so that they don't have to share account passwords if they want somebody else to be able to publish to the site using more standard tooling, and other means of being able to poll
the website in a manner which doesn't end up with the site owner sending legal threats because you wanted to make it possible for somebody to query the website, but it redirects to the Discord pages for each plugin, except that Discord doesn't allow you
to have more than 100 servers (except for those who can afford to waste their money on Nitro...). The reality is that forums are basically dead; this has been a long-standing trend on the wider internet, for the worse. Between growing up, meaning that I no longer have the leisure of browsing a forum all day as I did 10 years ago, and the fact that
Discord is just there, it's pretty hard for many to justify pursuing the forums when they can have more active conversations on Discord. Why? Because developers are bored and want to turn a 1950s car into a post-2000s prototype. Bored? The situation was unmanageable. We were unable to browse our patches folder. Merging in PRs was an entire
bunch of operations due to how many patches we were trying to maintain that the entire process of merging in a PR was an entire bunch of steps in which often resulted in moans because that one trivial PR you merged made it impossible to merge in this other PR we needed without another half an hour dedicated to this, resulting in multiple hour
dedicated to the PR queue only to be able to merge in a handful of patches. Because developers are bored and want to turn a 1950s car into a post-2000s prototype. All we want is for our servers to actually work. If we want is for our servers to actually work. If we want to turn a 1950s car into a post-2000s prototype. All we want is for our servers to actually work. If we want to turn a 1950s car into a post-2000s prototype.
existing as a project whose goal is to improve performance and expand upon API if we're not going to expand upon the API and support the things that developers ask us to provide API for? We might as well just throw the towel in, archive the repo, and close the discord/forums. Paper forgets where it comes from, its origins, and what made its
existence possible. Paper was a little fork maintained by a group of people because they found it fun to do so. We continued to expand over the years because we added features and an API that people wanted, which we felt would be worth the effort to maintain. We never intended to get big; this was generally just a fun project we enjoyed sinking
time into, wondering if one day we'd ever be able to break away from Spigot and start working on the things we wanted to do and that plugin developers and server owners wanted us to do. The project started with two people and grew as people wanted to do and that plugin developers and server owners wanted us to do. The project started with two people and grew as people wanted to do and that plugin developers and server owners wanted us to do.
point where spigots areas of "no interestin supporting" pretty much covers the direction that mojang was steering into for the past decade, and many of our decisions to not work too much on the stuff spigot refused to support has consistently bit us. Even before the hard fork, we'd started to lose compatability with spigot plugins due to them once in a
while working on API they'd said no to in the past, resulting in many occasions where we had to refuse to do the thing we set out to do out of concern that spigot would do something that would break our ability to remain compatible with them. Reactions: About of these plugins only still work due to Spigots attitude towards never breaking
plugins ever (except for the random coin toss in which they broke API in manners that could cause duplication issues or just crash servers with 0 warning mid-release cycle), to the degree that they invested the effort into a chonky mechanism which rewrites the code of every single plugin that loads, ever; and have often refused to expose features to
plugins in order to prioritise legacy plugins over being able to do basic operations like query the biome of a chunk. They finally accepted that this would harm their ability to move forward and eventually began working on some of the things the developer community had been asking them to do for a decade. The bytecode rewriting stuff is sadly accepted that this would harm their ability to move forward and eventually began working on some of the things the developer community had been asking them to do for a decade.
 something the team has decided to carry on with going forwards in order to let devs pretend that mojang isn't moving the ecosystem forward. At the time, that was a goal we considered beneficial for both the community and ourselves. In good faith, we cannot refuse to move forward with the Minecraft ecosystem. We either adopt Mojang's efforts to
improve the power available to server owners or give up and close shop. We have consistently talked about our plans to break away from spigot for years now, in public, there is no tricking here, we just finally decided we could no longer remain in our current situation and decided that it was time to deliver on that promise. Reasonable? Are you
paying us for this extra effort that would double, if not triple, the amount of work we're doing? A large part of the reason why we hard forked was becoming increasingly unbearable; The current update has some things that we need to look into improving, such as the bus factor of various
dependencies, but it was nice to be able to actually look into writing API for stuff that's needed going forward. This was only going to increasingly become an issue as mojang starts working more on various things, such as making it possible for servers to define custom Items, and as they work on their data component system, something which spigo
seemingly has 0 intent of actually supporting properly. We're only planning to expose more features towards the API in order to allow them to do things that require API breaks in order to pull off, such as, when mojang delivers on the framework, the support for custom items, something which is literally impossible to handle using the currently
supported API from upstream. Hangar is generally a work in progress, and many other platforms are out there, as noted. There is also places for feedback on suggestions and improvements, though, some aspects, like commands, are a bit of a concern more in terms of policy (i.e. do we try to manage aspects of moderation that spigot refuses to deal
with, or, do we just leave it to be an unmoderated area outside of stuff we are morally/legally required to deal with). I do not know a single developer who consistent complaints and feature requests against that system, which Spigot is unable to
fulfil. it's a Forum running a plugin that allows people to list stuff and doesn't support any basic feature that exists within the rest of the entire ecosystem that devs have been begging for for years, such as the ability to have teams so that they don't have to share account passwords if they want somebody else to be able to publish a plugin, efforts on
the search tool to allow people to find stuff; automated access for plugin devs to be able to publish to the site using more standard tooling, and other means of being able to publish to the website in a manner which doesn't end up with the site owner sending legal threats because you wanted to make it possible for somebody to query the website using a
well designed API. The reality is that forums are basically dead; this has been a long-standing trend on the wider internet, for the worse. Between growing up, meaning that I no longer have the leisure of browsing a forum all day as I did 10 years ago, and the fact that Discord is just there, it's pretty hard for many to justify pursuing the forums when
they can have more active conversations on Discord. Bored? The situation was unmanageable. We were unable to browse our patches we were trying to maintain that the entire process of merging in a PR was an entire bunch of steps in which often resulted in moans
because that one trivial PR you merged made it impossible to merge in this other PR we needed without another half an hour dedicated to this, resulting in multiple hour dedicated to the PR queue only to be able to merge in a handful of patches. Mojang is moving forward, more so these days. What is the point of existing as a project whose goal is to
improve performance and expand upon API if we're not going to expand upon the API and support the things that developers ask us to provide API for? We might as well just throw the towel in, archive the repo, and close the discord/forums. Paper was a little fork maintained by a group of people because they found it fun to do so. We continued to
expand over the years because we added features and an API that people wanted, which we felt would be worth the effort to maintain. We never intended to get big; this was generally just a fun project we enjoyed sinking time into, wondering if one day we'd ever be able to break away from Spigot and start working on the things we wanted to do and
that plugin developers and server owners wanted us to do. The project started with two people and grew as people wanted a place where they could work on things spigot had no interest in supporting, we've come to the point where spigots areas of "no interest in supporting" pretty much covers the direction that mojang was steering into for the past
decade, and many of our decisions to not work too much on the stuff spigot refused to support has consistently bit us. Even before the hard fork, we'd started to lose compatability with spigot plugins due to them once in a while working on API they'd said no to in the past, resulting in many occasions where we had to refuse to do the thing we set out
to do out of concern that spigot would do something that would break our ability to remain compatible with them. Thank you for your comprehensive response I won't bother you anymore; I'll see what the future holds... Despite my concerns and criticisms, I thank you for all the work you've done over the years on Paper. I just hope I can continue the
adventure with you. Have a good Sunday, everyone! Updates to Paper do not have any sort of estimate for when they release, ever. Any and all updates will arrive when they are ready, and the only thing to do is wait for them patiently along with everyone else. Reactions: naive and TOWUK Your server should work fine for Bedrock 1.21.70 players if
the server's on 1.21.4 and is using the latest build of Geyser. There's no solution yet for Java players on 1.21.5. No I'm having the same issue. I tried viaversion and it didn't work. In the server's on 1.21.4 and is using the latest build of Geyser. There's no solution yet for Java players on 1.21.5. Wo I'm having the same issue. I tried viaversion and it didn't work. In the server's on 1.21.4 and is using the latest build of Geyser. There's no solution yet for Java players on 1.21.5.
know there's a spigot 1.21.5 out but I don't know how to replace my server with that one. We don't recommend running Spigot, especially this early in release cycle. Lots of potential bugs to iron out. If you need help with geyser, see geyser's support. For viaversion, see viaversion's support. I know via has an update for 1.21.5, but can't speak for
geyser. We don't recommend running Spigot, especially this early in release cycle. Lots of potential bugs to iron out. If you need help with geyser, see geyser's support. For viaversion, see viaversion, see viaversion and Geyser are separate plugins.
Paper produces neither, but both run great on it. ViaVersion and Geyser are separate plugins. Paper produces neither, but both run great on So I'll just have to wait till paper 1.21.5 comes out? or do you know how to change it to spigot cause I'm already running it on there for performance. ViaVersion would enable 1.21.5 players to play on your
server. Geyser would enable bedrock players. I would not recommend running Spigot instead of those things, but if you wish to run it that is not something we provide support for 1.21.4, however. We do not
provide support for server software that isn't ours, noting that running your server on Spigot might cause damage to existing data on your server and used by plugins, you would really be better off waiting until Paper has releases, rather than running early experimental spigot builds. Thank you, I just messaged bedrock players of my server and
apologized for the inconvenience. While you're hear, does hydraulic work? If you don't know it's a plugin that allows bedrock players to use mods. We do not produce geyser or hydraulic. You should ask them for help with their stuff. Paper 1.21.4 + ViaVersion + the latest build of Geyser works fine for me for 1.21.70 players, but I don't think
ViaVersion is making any difference since it doesn't support 1.21.5 yet. Not sure what's going on for you. Paper 1.21.4 + ViaVersion is making any difference since it doesn't support 1.21.5 yet. Not sure what's going on for you. Do you have aditional plugins
ViaVersion has early releases for 1.21.5, as already said. Paper 1.21.4 + ViaVersion + the latest build of Geyser works fine for me for 1.21.5 yet. Not sure what's going on for you. Are you also talking about bedrock players on console? What? I was asking if
the person with a working server had additional plugins. And that person claimed ViaVersion doesn't have an update. Remember, this place isn't the place for support with bedrock clients or with viaversion. You can get better support the place isn't the p
place for support with bedrock clients or with viaversion. You can get better support directly from those folks! Oh, i'm just confused I guess. My plugin list: BedWars-0.2.36.1.jar CommandBlocks-v1.4.0.jar EssentialsXSpawn-2.21.0.jar EssentialsXSpawn-2.2
LuckPerms-Bukkit-5.4.157.jar PlaceholderAPI-2.11.6.jar ProtocolLib.jar PurpurExtras-1.34.5.jar SkinsRestorer.jar TAB v5.0.7.jar VaultUnlocked-2.9.0.jar WiaBackwards-5.2.1.jar ViaVersion-5.2.1.jar Vi
portals-4.2.3.jar playeronlyplates-1.0.1-SNAPSHOT.jar worldedit-bukkit-7.3.11.jar worldedit-bukkit-7.0.13-dist.jar I also have MCXboxBroadcast update for console to join. On it specifies that 1.21.40-70 works fine on a 1.21.4
server. Updates to Paper do not have any sort of estimate for when they release, ever. Any and all updates will arrive when they are ready, and the only thing to do is wait for them patiently along with everyone else. It seems you are a "staff member"? Isn't kindness free to use? A new user asked about timing of release without being demanding, and
this is how you respond? Saltiness is not required. I realize the buzz and demands made around version releases can be rude and taxing, however, this isn't the case in this instance. I'm a new user to Paper and your forum, and I wouldn't expect to be responded to like this. How 'bout make mention of a reasonable expectation of time for release but
that it isn't something that can be guaranteed, or maybe how long the last updates took to make available? That would be useful for a new user to hear. It seems you are a "staff member"? Isn't kindness free to use? A new user asked about timing of release without being demanding, and this is how you respond? Saltiness is not required. I realize the
buzz and demands made around version releases can be rude and taxing, however, this isn't the case in this instance. I'm a new user to Paper and your forum, and I wouldn't expect to be responded to like this. How 'bout make mention of a reasonable expectation of time for release but that it isn't something that can be guaranteed, or maybe how long
the last updates took to make available? That would be useful for a new user to hear. On our Discord, we have found this message best communicates expectations in a concise manner (users tend to stop reading by the 3rd sentence) and is
easier than writing up a fresh response each time. As for your thoughts on reasonable expectations: This project is run by volunteers, and their time is variable. Minecraft updates can greatly vary in terms of complexity, both in terms of actual internal changes and in terms of changes we have to make to adapt within our API. We just recently
migrated our tooling to be independent of Spigot (see the hard fork announcement) and are working out the kinks in our setup. As a result of these various factors, it is never possible to provide an ETA (and especially with the current situation). Updates have been as quick as pretty much same-day and as long as many, many weeks. Your patience is
appreciated. Reactions: EverPilgrim that is just a prewritten bot rely we have used for ages and not ones has somebody complained in the thousands of times it was used. we will never give any indication of time, ever, since a) every update is different, so its impossible to predict based on last updates and b) people would complain even more if that
estimate wasn't accurate, just wait patiently like everybody else, an announcement will be made when it is available. I'm excited, too! I hope it's out soon. Page 2 My plugin list: BedWars-0.2.36.1.jar CommandBlocks-v1.4.0.jar EssentialsXSpawn-
2.21.0.jar Geyser-Spigot.jar LuckPerms-Bukkit-5.4.157.jar PlaceholderAPI-2.11.6.jar ProtocolLib.jar PurpurExtras-1.34.5.jar SkinsRestorer.jar TAB v5.0.7.jar VaultUnlocked-2.9.0.jar ViaBackwards-5.2.1.jar ViaVersion-5.2.1.jar ViaVersion-5.2.
netherportals-4.2.3.jar multiverse-portals-4.2.3.jar multiverse-portals-4.2.3.jar multiverse-portals-4.2.3.jar playeronlyplates-1.0.1-SNAPSHOT.jar worldguard-bukkit-7.0.13-dist.jar I also have MCXboxBroadcast in Geyser's extension folder. Bedrock PC/mobile/console on 1.21.40-70 all works fine, but I needed the MCXboxBroadcast update for console to join. On it specifies that
1.21.40-70 works fine on a 1.21.4 server. Hi! So sorry to ask and this might be slightly off topic, but how did you get CommandBlocks to work? I cannot use it for the life of me and there are no tutorials, paper wake up the new version is here A User Who Deleted Their Own Post said: It's a good thing customer service isn't needed for your free
jerk interpretation from, but it's sad to hear. The amount of hours the team contributes every single day, helping users and developers on our Discord (our most active platform for support), is unreal. And they do it while being friendly and welcoming of anyone from newbies to highly experienced users/devs, because they are good people. I hope you
 join us on Discord and see for yourself. Reactions: Entity 616 where can we get information about the advancement of the update? We post update when we have them on our Discord server in the update can we get information about the advancement of the update when we have them on our Discord server in the update when we have them on our Discord server in the update can we get information about the advancement of the update when we have them on our Discord server in the update can we get information about the advancement of the update when we have them on our Discord server in the update when we have them on our Discord server in the update when we have them on our Discord server in the update when we have them on our Discord server in the update when we have the update when update when we have the update when upda
builds have been released! As always, backups are absolutely mandatory. After upgrading your world to 1.21, you cannot downgrade back to a lower version! We would like to support PaperMC as a whole, you can find more information at . spark profiler inclusion in PaperThanks to riley and
Luck, Paper now bundles spark as its main profiler for diagnosing causes of lag. Timings is now disabled by default and will be entirely removed at a later date, possibly with 1.22. As a developer, please make sure you remove any custom Timing uses by then. You can see our docs page as well as the GitHub Discussions page for more details and also
provide feedback there. Configuration changesThe disable-teleportation-check config option is now gone, as this check no longer exists in vanilla. We have also changed a number of default configuration values to improve the gameplay experience: merge-radius.exp: 3->-1(disabled). vanilla has its own less aggressive merging logic, but
 generally xp orbs were not a large performance concern; otherwise we might instead offer new options to change vanilla's merge logic merge-radius.item: 2.5->0.5, reflecting the vanilla default, since the increased value was often seen as disruptive. If you expect large number of the same item types to be lying around close to one another, you can
 increase this again entity-activation-range.raider: 48->64 Various values under entity-tracking-range have been increased to make sure you can actually see monsters and players attacking you from farther away (e.g. no longer running into invisible ghasts shooting at you in the nether. Note that this does not affect ticking, only whether they are sent
to a player) players: 48->128 animals: 48->96 monsters: 48->96 misc: 32->96 Slower than usual startup performance, adding multiple seconds to each individual larger plugin. This has unfortunately been made slightly worse again in 1.21.^1 We have already
mildly improved on it and are working on reducing it by as much as possible, but in the meantime you can work around it by disabling the cross-version compatibility measures by either optionally using Paper's plugin loader as a developer, or by using the paper startup flag. However, the flag will only work if all of your plugins loader as a developer, or by using the paper startup flag.
are built against the latest API version. 'Done'-message changesThe final Done (7.392s)! For help, type "help" message now shows the time from when the Minecraft server initially bootstrapped. Previously, it was in a kind of weird spot where it only tracked world loading, plugins, and a few other parts of server startup. We have also reinstated
vanilla's original Done preparing level message next to the total startup time. The .paper-remapped directory caused a bit of confusion, so
here's what it does: It stores the remapped plugin jars as well as a cached server jar, so that all of these don't need to (and generally shouldn't) touch it. For developers In case you skipped the 1.20.5/6 update, make sure to read its
announcement on Mojang mappings use at runtime and our new Brigadier command API. Attribute modifiers no longer have a name and methods are unusable as of now, only the Paper-added get/removeModifier via unids have had a
 temporary compatibility measure put in. If you are using this API, make sure to move to their replacements as soon as possible. Removed chunk gen delegation and regeneration methodsOur vanilla chunk gen delegation approaches the replacements as soon as possible. Removed chunk gen delegation and regeneration methodsOur vanilla chunk gen delegation approaches the replacements as soon as possible.
implementation is no longer feasible due to how hard-coupled and stateful a lot of the handling has become. The only way to properly regenerate chunks. ItemStack and ItemMeta (again) The ItemStack class within the API module no
longer holds its method implementations directly, but instead always redirects to a held internal (Craft)ItemStack. If this affects you and you aren't already using a Minecraft testing framework, we recommend using something
like Mockbukkit to mock a running server instance. We have added ItemMeta#hasDamageValue to check whether the damage item data component has been added to an item. hasDamage will still return whether there is a non-0 amount of damage. For resetting damage, we recommend using resetDamage instead of setting it to 0 to improve item
comparison. There's still other issues thanks to ItemMeta clashing hard with Minecraft's new item data component API in the background and will let you know more once it has been merged. Other changes
End gateway teleportation cancellation doesn't reset the portal cooldown, so you should use EntityPortalEnterEvent with PortalType.END GATEWAY to check for initial entries Projectile#getWeapon is now nullable Opportunity to change the git commit author details in Paper commitsIf you have previously contributed to our main Paper repository or
GitHub and want the email or name that was used on that commit to be changed, you may use the modmail command on our Discord to tell us the new details. As per our planned repository restructure in the future, the current git history will be overridden, so we thought we might as well fix the details for anyone that needs it. 1 Here are a few
examples (results will vary depending on hardware and OS), which will quickly add up the more plugins you have: ViaVersion takes over two seconds longer (from a bout 1.5 to 4 seconds) CoreProtect takes a second and a half longer (on a clean setup with
otherwise no measurable load time at all if rewriting is disabled) Last edited by a moderator: Jul 20, 2024 Reactions: cat, ysl3000, LoJoSho and 1 other person Messages 27 Reaction score 150 Points 3 Page 2 Stable Paper and Velocity 1.21 builds have been released! As always, backups are absolutely mandatory. After upgrading your world to 1.21
you cannot downgrade back to a lower version! We would like to thank everyone that worked on this update:If you'd like to support PaperMC as a whole, you can find more information at . spark profiler inclusion in PaperThanks to riley and Luck, Paper now bundles spark as its main profiler for diagnosing causes of lag. Timings is now disabled by
default and will be entirely removed at a later date, possibly with 1.22. As a developer, please make sure you remove any custom Timing uses by then. You can see our docs page as well as the GitHub Discussions page for more details and also provide feedback there. Configuration changesThe disable-teleportation-suffocation-check config option is
now gone, as this check no longer exists in vanilla. We have also changed a number of default configuration values to improve the gameplay experience: merge-radius.exp: 3->-1 (disabled). vanilla has its own less aggressive merging logic, but generally xp orbs were not a large performance concern; otherwise we might instead offer new options to
change vanilla's merge logic merge-radius.item: 2.5->0.5, reflecting the vanilla default, since the increased value was often seen as disruptive. If you expect large number of the same item types to be lying around close to one another, you can increase this again entity-activation-range.radius.item: 2.5->0.5, reflecting the vanilla default, since the increased value was often seen as disruptive. If you expect large number of the same item types to be lying around close to one another, you can increase this again entity-activation-range.radius.item: 2.5->0.5, reflecting the vanilla default, since the increased value was often seen as disruptive. If you expect large number of the same item types to be lying around close to one another, you can increase this again entity-activation-range.radius.item: 2.5->0.5, reflecting the vanilla default, since the increased value was often seen as disruptive. If you expect large number of the same item types to be lying around close to one another, you can increase this again entity-activation are also as a second contract of the same item types to be lying around close to one another, you can increase this again entity-activation are also as a second contract of the same item types to be lying around close to one another, you can increase the same item types to be lying around close to one another are also as a second contract of the same item types to be lying around close to one another are also as a second contract of the same item types to be lying around close to one another are also as a second contract of the same item types to be lying around close to one another are also as a second contract of the same item types to be lying around close to one another are also as a second contract of the same item types to be lying around close to one another are also as a second contract of the same item types to one another are also as a second contract of the same item types to one another are also as a second contract of the same item types to one another are also as a second 
been increased to make sure you can actually see monsters and players attacking you from farther away (e.g. no longer running into invisible ghasts shooting at you in the nether. Note that this does not affect ticking, only whether they are sent to a players attacking you from farther away (e.g. no longer running into invisible ghasts shooting at you in the nether. Note that this does not affect ticking, only whether they are sent to a player) players: 48->96 misc: 32->96 misc: 32->96 Slower than usual startupIr
1.20.5/6, Spigot has made additions to their plugin rewriting that resulted in poor startup performance, adding multiple seconds to each individual larger plugin. This has unfortunately been made slightly worse again in 1.21.1 We have already mildly improved on it and are working on reducing it by as much as possible, but in the meantime you can
work around it by disabling the cross-version compatibility measures by either optionally using Paper's plugin loader as a developer, or by using the paper.disableOldApiSupport startup flag. However, the flag will only work if all of your plugins are built against the latest API version. 'Done'-message changesThe final Done (7.392s)! For help, type
"help" message now shows the time from when the Minecraft server initially bootstrapped. Previously, it was in a kind of weird spot where it only tracked world loading, plugins, and a few other parts of server startup. We have also reinstated vanilla's original Done preparing level message next to the total startup time. The .paper-remapped directory
As per the last announcement, we now use Mojang mappings at runtime and thus have to remap plugins that might still be using Spigot's mess of mappings. The .paper-remapped folder in the plugin directory caused a bit of confusion, so here's what it does: It stores the remapped plugin jars as well as a cached server jar, so that all of these don't
have to be processed during every single server startup. The folder is automatically cleaned up, so you don't need to (and generally shouldn't) touch it. For developers In case you skipped the 1.20.5/6 update, make sure to read its announcement on Mojang mappings use at runtime and our new Brigadier command API. Attribute modifiers Attribute
modifiers no longer have a name and uuid and instead make use of a single string key as its identifier. The old constructors and methods are unusable as of now, only the Paper-added get/removeModifier via uuids have had a temporary compatibility measure put in. If you are using this API, make sure to move to their replacements as soon as possible
properly regenerate chunks with structures and everything else attached is to generate a world with the same seeds and to copy over those chunks. ItemStack and ItemMeta (again)The ItemStack and ItemMeta (again)The ItemStack and ItemMeta (again)The ItemStack and ItemMeta (again)The ItemStack and 
This won't matter to most people, but it means that you can no longer run unit tests using the API-only ItemStack. If this affects you and you aren't already using a Minecraft testing framework, we recommend using something like Mockbukkit to mock a running server instance. We have added ItemMeta#hasDamageValue to check whether the
damage item data component has been added to an item. hasDamage will still return whether there is a non-0 amount of damage, we recommend using resetDamage instead of setting it to 0 to improve item comparison. There's still other issues thanks to ItemMeta clashing hard with Minecraft's new item data storage, including
unfortunate but not really solvable behavioral breaks to item flags. Right now, we're still working on our improved item data component API in the background and will let you know more once it has been merged. Other changes End gateway teleportation cancellation doesn't reset the portal cooldown, so you should use EntityPortalEnterEvent with
PortalType.END GATEWAY to check for initial entries Projectile#getWeapon is now nullable Opportunity to change the git commit author details in Paper commitsIf you have previously contributed to our main Paper repository on GitHub and want the email or name that was used on that commit to be changed, you may use the modmail command on
our Discord to tell us the new details. As per our planned repository restructure in the future, the current git history will be overridden, so we thought we might as well fix the details for anyone that needs it. 1 Here are a few examples (results will vary depending on hardware and OS), which will quickly add up the more plugins you have: ViaVersion
takes over two seconds longer to load (from a few hundred ms to 2.5 seconds) WorldEdit takes two and a half seconds longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (on a clean setup with otherwise no measurable load time at all if rewriting is disabled) Last edited by a moderator: Jul 20, 2024 Reactions: cat
ysl3000, LoJoSho and 1 other person Messages 27 Reaction score 150 Points 3 Page 3 Stable Paper and Velocity 1.21 builds have been released! As always, backups are absolutely mandatory. After upgrading your world to 1.21, you cannot downgrade back to a lower version! We would like to thank everyone that worked on this update:If you'd like to
support PaperMC as a whole, you can find more information at . spark profiler inclusion in PaperThanks to riley and Luck, Paper now bundles spark as its main profiler for diagnosing causes of lag. Timings is now disabled by default and will be entirely removed at a later date, possibly with 1.22. As a developer, please make sure you remove any
custom Timing uses by then. You can see our docs page as well as the GitHub Discussions page for more details and also provide feedback there. Configuration changes the disable-teleportation-check configuration values to
improve the gameplay experience: merge-radius.exp: 3->-1(disabled). vanilla has its own less aggressive merging logic, but generally xp orbs were not a large performance concern; otherwise we might instead offer new options to change vanilla's merge logic merge-radius.item: 2.5->0.5, reflecting the vanilla default, since the increased value was
often seen as disruptive. If you expect large number of the same item types to be lying around close to one another, you can increased to make sure you can actually see monsters and players attacking you from farther away (e.g. no longer
running into invisible ghasts shooting at you in the nether. Note that this does not affect ticking, only whether they are sent to a player) players: 48->96 misc: 32->96 slower than usual startup. 1.20.5/6, Spigot has made additions to their plugin rewriting that resulted in poor startup performance, adding multiple
 seconds to each individual larger plugin. This has unfortunately been made slightly worse again in 1.21.¹ We have already mildly improved on it and are working on reducing it by as much as possible, but in the meantime you can work around it by disabling the cross-version compatibility measures by either optionally using Paper's plugin loader as a
developer, or by using the paper.disableOldApiSupport startup flag. However, the flag will only work if all of your plugins are built against the latest API version. 'Done'-message changesThe final Done (7.392s)! For help, type "help" message now shows the time from when the Minecraft server initially bootstrapped. Previously, it was in a kind of
weird spot where it only tracked world loading, plugins, and a few other parts of server startup. We have also reinstated vanilla's original Done preparing level message next to the total startup time. The .paper-remapped directoryAs per the last announcement, we now use Mojang mappings at runtime and thus have to remap plugins that might still
be using Spigot's mess of mappings. The .paper-remapped folder in the plugin directory caused a bit of confusion, so here's what it does: It stores the remapped folder is automatically cleaned up, so you don't need to (and
generally shouldn't) touch it. For developers In case you skipped the 1.20.5/6 update, make sure to read its announcement on Mojang mappings use at runtime and our new Brigadier command API. Attribute modifiers no longer have a name and unid and instead make use of a single string key as its identifier. The old constructors
and methods are unusable as of now, only the Paper-added get/removeModifier via uuids have had a temporary compatibility measure put in. If you are using this API, make sure to move to their replacements as soon as possible. Removed chunk gen delegation and regeneration methodsOur vanilla chunk gen delegation API already broke with almost
every major update and this one is no exceptions. Together with the regenerate chunks with structures and everything else attached is to generate a world with the same seeds and to
copy over those chunks. ItemStack and ItemMeta (again)The ItemStack class within the API module no longer holds its method implementations directly, but it means that you can no longer run unit tests using the API-only ItemStack. If this
affects you and you aren't already using a Minecraft testing framework, we recommend using something like Mockbukkit to mock a running server instance. We have added to an item. hasDamage will still return whether there is a non-0 amount of
damage. For resetting damage, we recommend using resetDamage instead of setting it to 0 to improve item comparison. There's still other issues thanks to ItemMeta clashing hard with Minecraft's new item data storage, including unfortunate but not really solvable behavioral breaks to item flags. Right now, we're still working on our improved item
data component API in the background and will let you know more once it has been merged. Other changes End gateway teleportation cancellation doesn't reset the portal cooldown, so you should use EntityPortalEnterEvent with PortalType.END GATEWAY to check for initial entries Projectile#getWeapon is now nullable Opportunity to change the
git commit author details in Paper commits f you have previously contributed to our main Paper repository on GitHub and want the email or name that was used on that commit to be changed, you may use the modmail command on our Discord to tell us the new details. As per our planned repository restructure in the future, the current git history will
 be overridden, so we thought we might as well fix the details for anyone that needs it. 1 Here are a few examples (results will vary depending on hardware and OS), which will quickly add up the more plugins you have: ViaVersion takes over two seconds longer to load (from a few hundred ms to 2.5 seconds) WorldEdit takes two and a half secondsing on hardware and OS), which will quickly add up the more plugins you have: ViaVersion takes over two seconds longer to load (from a few hundred ms to 2.5 seconds) WorldEdit takes two and a half secondsing or hardware and OS), which will quickly add up the more plugins you have: ViaVersion takes over two seconds longer to load (from a few hundred ms to 2.5 seconds) WorldEdit takes two and a half secondsing or hardware and OS), which will quickly add up the more plugins you have: ViaVersion takes over two seconds longer to load (from a few hundred ms to 2.5 seconds) WorldEdit takes two and a half secondsing takes over two seconds longer to load (from a few hundred ms to 2.5 seconds) WorldEdit takes two and a half secondsing takes over two seconds longer to load (from a few hundred ms to 2.5 seconds) WorldEdit takes two and a half secondsing takes over two seconds longer to load (from a few hundred ms to 2.5 seconds) WorldEdit takes two and a half secondsing takes over two seconds longer to load (from a few hundred ms to 2.5 seconds).
longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (on a clean setup with otherwise no measurable load time at all if rewriting is disabled) Last edited by a moderator: Jul 20, 2024 Reactions: cat, ysl3000, LoJoSho and 1 other person Messages 27 Reaction score 150 Points 3 Known Issues None! All known issues have
been fixed. Please report new issues on github Previous issues (fixed) Spoiler: Does not work for all dupers Currently only this type of sand duper (pictured) can work with this plugin. This is most sand duper, including traditional anvil and dragon egg duper. Sand duper using cobble wall or boats will not work. I do not have good way to detect them,
ask paper developer and they are not sure either but if you have idea (NMS OK) please tell me or submit pull request. I have tried very hard but I am not sure it is possible without bootstrap for mixin or other class transformation. Last edited by a moderator: Feb 13, 2022 hey, this is great, one tiny problem I had was that it kept deleting my pistons
that were under the end portal. i don't know how to fix it. hey. this is great. one tiny problem I had was that it kept deleting my pistons that were under the end portal. i don't know how to fix it. This isn't something that gravitycontrol will cause. It does not delete or overwrite any blocks. Please upload a screenshot/schematic/world download for
replication of this issue. Does this work in 1.19.2? It's been a hot minute since the last post, but here we are! As always, backups are absolutely mandatory. After upgrading your world to 1.21.7, you cannot downgrade back to a lower version! We can also give you a pinky promise that it won't take this long to exit the experimental phase again and weep are!
have already addressed this inflexibility in build channels - more on below and on our Discord. We would like to support PaperMC as a whole, you can find more information at . Among other changes and the new V3 of our downloads API, its release channels have changed to: alpha, beta,
stable, recommended. alpha: Equivalent to the prior experimental label beta: New middle-point for builds that aren't entirely unstable, but partially unfinished (such as missing datafixer changes) stable: Equivalent to the prior default label - always make sure you stay up-to-date with new builds
here, as important fixes and changes continue being pushed recommended: Unused in Paper, but currently used for Velocity releases. If this ever changes, we will announce it beforehand Time and time again, issues in old world upgrade code are found, with fairly important issues having been fixed by Mojang or Paper in the recent versions. If you
use the force-upgrade startup flag, you lose out on fixes from future versions given it force-upgrade on 1.21.7 Paper (!), but you
should otherwise never force upgrade without a specific reason. For many years, the Bukkit reload mechanic has been unofficially deprecated, with big warnings whenever some triggers it via /reload command will now point to the /minecraft:reload
command, a reload mechanic that is and will stay supported, as it's a Vanilla mechanic. Users should be using hot reloading, or doing actual restarts as well. For guidance on how to hotswap, see the pinned messages in our paper-dev Discord
channel. For developers The use of PlayerLoginEvent is now deprecated. This change has been made to allow us to start developing API for the Configuration Phase in Minecraft. This most notably allows for dialoges to be sent to the player before they join the game, or for resource packs to be sent before as well. This is an early warning. While
PlayerLoginEvent will remain available for now, breaking changes may be introduced in the future. Plugins relying on this event should be able to join events based on your needs: PlayerConnectionValidateLoginEvent This should be used when wanting to control if a player should be able to join events based on your needs.
the game. Essentialy, any prevention of the player joining logic that was previously done in the PlayerLoginEvent. Note that this is ran two times, first when logging in, and then when exiting the configuration phase. So, you do not have access to a Player entity, only a PlayerProfile. PlayerServerFullCheckEvent A common use of the PlayerLoginEvent
was to allow players to join the server even if it was full, this event allows you to override the fullness check for each player. The configuration phase has been added between logging into a server and fully joining a world. Here, server data, including tags and custom registry elements are sent to the client. It is in this phase that you send custom data, including tags and custom registry elements are sent to the client. It is in this phase that you send custom data, including tags and custom registry elements are sent to the client. It is in this phase that you send custom data, including tags and custom registry elements are sent to the client. It is in this phase that you send custom registry elements are sent to the client.
to a client or where you an apply a resource pack before they exit the loading screen. On top of that, you can throw a player Configuration exign the configuration phase using Player pack before they exit the loading screen. On top of that, you can throw a player back into the configuration phase using Player back into the configuration phase using Player pack before they exit the loading screen. On top of that, you can throw a player back into the configuration phase using Player pack before they exit the loading screen.
directly using ClickEvent.openDialog. For custom clicks actions, you may use the PlayerCustomClickEvent, but the better and preferred option is to directly pass a consumer-like callback in DialogAction.customClick, which will be run when the player clicks the specified element. ItemStacks now use the SNBT format when writing to configuration
files. This allows us to run ItemStack upgrades through Minecraft's data fixers, going through much more reliable upgrades of existing stacks. Direct serialization of ItemMeta (as in, storing ItemMeta in a config) is now considered unsupported and will likely be removed in the future. If you want to serialize items to other formats, such as JSON or as
before raw bytes, the (de)serializeStack methods on UnsafeValues might be of interest to you. Cow inheritance has been adjusted to fit Vanilla: MushoomCow no longer extends Cow, but the new AbstractCow Cow now extends Abstract
now have to use those new entity types instead and will no longer be able to swap out an lingering for a splash potion and vice versa without creating a new entity. Here is a small selection of recent API additions that might be interesting to you: Registry events for modifying/adding custom cat, chicken, cow, frog, pig, and wolf variants, as well as
damage types and paintings Player#openvirtualSign to open unplaced signs. Changes can be checked via UncheckedSignChangeEvent Entity#setCooldown(Key, int ticks) and its getter for grouped cooldowns rather than single
Server#sendRichMessage(String) and sendPlainMessage for broadcasting and logging plain/minimessage text Other new events, including PlayerMapFilledEvent, PlayerPickEntityEvent, VaultChangeStateEvent, ClientTickEndEvent, EntityEquipmentChangedEvent, EntityAttemptSmashAttackEvent, VaultChangeStateEvent, ClientTickEndEvent, PlayerPickEntityEvent, VaultChangeStateEvent, ClientTickEndEvent, EntityEquipmentChangedEvent, EntityAttemptSmashAttackEvent, VaultChangeStateEvent, VaultChangeStateEvent, VaultChangeStateEvent, ClientTickEndEvent, PlayerPickEntityEvent, VaultChangeStateEvent, VaultChangeSt
PlayerClientLoadedWorldEvent, EntityEffectTickEvent After the release of the first builds for Minecraft 1.21.4, we are happy to share some even more exciting news with everyone. Following the successful rollout of our Mojang-mapped server in 1.20.5, we are taking a big next step for the project: Since the project is inception, Paper has been built on
top of Spigot, consistently staying up-to-date with features introduced to it. Today, Paper applies nearly 1600 additional patches with spigot has been limiting the project, most noticeably with slower version updates since its updates to
snapshots, pre-releases and release candidates are worked on behind closed doors. Hardforking removes this unnecessary delay and sees Paper become its own independent project, streamlining development of the server and major missing API, such as our recent registry and item data component API. This does not mean that existing
configs/API/behavior/etc. will be removed. Starting with 1.21.4, Paper will simply no longer be bound to Spigot's future changes, enabling us and the community to move forwards on our own terms. While this may sound scary, we have bundled up some helpful disclaimers for each relevant user group down below. We are committed to making this
process as smooth as possible for everyone. If any more questions arise, feel free to ask us on our Discord! As a server owner/administrator, there will be no change initially. For now, both bukkit.yml and spigot.yml will continue to work how they always have. Plugins already running on the stable 1.21.4 builds as well as plugins compiling to older
versions of Spigot will also continue to run. As the plugin API of Paper and Spigot slowly diverge, the only potential worry may be plugins trying to use Spigot features introduced after the hardfork. We presume a large number of plugins trying to use Spigot features introduced after the hardfork.
upside for you is the considerably faster release speed of Paper during Minecraft releases, as mentioned prior. Post hardfork, we are able to update Paper to release-candidates, pre-releases and maybe even snapshots to get experimental builds out to everyone as early as possible. Just like for server owners, there will be no initial change in the plugin
API. Existing methods inherited from Spigot are maintained and will continue to work. This includes methods deprecated by us, like legacy text/chat color. However, we will no longer pull all new API added to Spigot after hardfork. To avoid accidentally calling such methods, we strongly suggest building your plugin against the Paper-API dependency.
We are still open to manually pulling some new upstream API into Paper to ensure plugin developers do not lose out on functionality, but you should not rely on it. Usage of server internals via paperweight-userdev will also continue to work as it has before, however, we recommend slowly preparing your plugin to run on a Mojang-mapped server with
iar and reflection remapping disabled. This can be achieved with the -Dpaper.disable PluginRemapping = true startup flag. If you are, you don't need to do anything else, as you can simply later remove the obfuscation step and will
be compatible with the future change - only reflective calls need to be addressed. However, we will give you ample time to prepare for this and will continue to automatically remap plugins for a while. Once that happens, it will become much easier to support multiple versions with little internal changes, which the obfuscated mappings and arbitrary
CraftBukkit package relocation currently prevent. The single set of standard mappings will also make debugging and code-sharing easier. With the switch to Paper-API and later a Mojang-mapped server, your plugins may no longer run on Spigot. Paper's market share of 85-90% on recent versions (according to some of the largest plugins on bStats,
including forks of Paper) should make it easy for you to support almost all of your users in the post-hardfork versions, even without being compatible with Spigot. Publishing plugins that require Paper can be done on numerous platforms, most notably Hangar and modrinth. Because we are merging in API/API-implementation source file history and the
new branch will have an entirely unrelated git commit history to the previous branches, we will eventually delete all older version branches, everything from ver/1.21.3. If you are in any way referencing them on our current repository, make sure to update their URL to: Our current compressed repo size is over 90MB - the new branch,
despite having many many more commits, will be less than 35MB in packed size, meaning cloning the repository will become much faster once the old branches have been removed. On top of that, looking at the previous patch-file history will also be a lot easier thanks to cleaning up index and line changes from the history. Hardfork affects
contributions to Paper dramatically, most of it for the better. With hardfork, the Paper repository will receive a full restructure, moving the entire API and API-implementation straight as source into the repository, Contributions to these can now directly be made on the java files, without the need to edit/rebuild patches. Changes to Mojang-owned
sources will still be based on patches, but each Vanilla source file will be represented by a single, per-file patch. For large changes, like anti-xray and moonrise, Paper will also offer the old feature-based patch approach on top of the previously described layout, combining the best of both techniques, next to an updated decompiler (Vineflower) and
mappings set (Parchment). In order to preserve history, we have merged multiple git trees, including our full patch-file history, and those patches applied into real commits over the existing Spigot history, and those patches applied into real commits over the existing Spigot history.
be closed later, and they cannot be force-pushed to target the new branch either. Please do not reopen PRs until we provide more detailed information on how to semi-automatically update old pull requests and have given the go-ahead for new ones. The main change you will have to deal with is the updated decompiler and mappings. We will provide
you with more information once our initial update process is finished. New versions of Paperweight patcher will be published to be compatible with our new repository. It's hard to give an exact timeline, but you will be able to follow the rough
progress of our hard fork process in the following GitHub issue: We will spend the following weeks updating the patch structure and fixing endless patch conflicts from the updated decompiler and new set of local/parameter mappings. Once that is done, we will push these builds as experimental builds to ensure we didn't
break any backwards compatibility or introduced any new issues. Once we have finished our tooling around forks and userdev, we will post an announcement on how to update forks and notify Paper contributors on how to properly update their pull requests to the new main branch. At some point down the line on new Minecraft releases, you can
expect API and preliminary builds to be published even before the full Minecraft release day. Eventually, we will start cleaning up and automatically migrating the different configs, and slowly remove long-disfunctional deprecated API, but our focus remains to make these transitions as smooth and graceful as possible. This change allows us to work
on snapshots, and to more freely work on major missing API. Contributions to Paper become a lot easier, and plugin development will become easier as well, both regarding API and server internals! Server admins: No action needed, but you might no longer be able to go back to Spigot starting with 1.21.4. Plugin development will become easier as well, both regarding API and server internals!
needed, but you should compile against Paper-API starting with 1.21.4. Paper contributors: You will have to redo your PRs, but we will tell you more about that later and provide you with useful scripts and tools. Forks/server hosts: Migrate any use of the old version branches (e.g. ver/1.8.8, ver/1.8.1.1; everything below 1.21.4) to the new archive
repository. Thank you for making the Paper organization into what it has become today, we're grateful that so many people choose to use our software! Focusing on preparations for this process took a toll on Paper and Velocity 1.21.3 builds are
out of the experimental phase! As always, backups are absolutely mandatory. After upgrading your world to 1.21.3, you cannot downgrade back to a lower version! We would like to thank everyone that worked on this update: If you'd like to support PaperMC as a whole, you can find more information at . After having added spark as our main profiler
for diagnosing causes of lag in 1.21, Timings has been set to no-op mode. This means that it can no longer be enabled or used, though its API classes will remain until a later update. As a developer, please make sure you remove any custom Timing uses by then. You can see our docs page as well as the GitHub Discussions page for more details and
also provide feedback there. Vanilla added a server properties option to pause world and entity ticking when no players are online after a while. This behavior is disabled by default on Paper because it is incompatible with what plugins expect and might do with no players online. You can enable it again by changing the value in the server properties
file, but we generally recommend against doing that unless you are 100% certain your plugins are compatible with server pausing or you may run into crashes or save data issues (they won't be able to properly work with entities and the world, or do other actions that would require "active" world ticking). Unlike the Bukkit schedulers, Folia's
GlobalRegionScheduler will not be ticked while the server is paused. If you ran a Spigot 1.21.3 server before switching back to Paper, we recommend manually setting pause-when-empty-seconds to -1 to disable it. Under entities spawning despawn-time, you can now configure hard despawn times in ticks for when an entity should be forcefully
despawned. An example usecase of this is preventing certain projectiles from being kept alive permanently. This patch was ported from Pufferfish with Kevin's go-ahead. We have added the legacy-ender-pearl-behavior config option to prevent ender pearls from being saved to the player and loading chunks, meaning they will behave like they did in
1.21.1 and before. Paper will default to the new vanilla behavior. We have fixed incorrect handling of damage reduction during invulnerable ticks after being hit to work like it does in Vanilla again. Mojang fixed string disarming behavior and its dupe, so we have dropped our patch and configuration option. For developers In case you skipped the
1.20.5/6 update, make sure to read its announcement on Mojang mappings use at runtime and our new Brigadier command API. As mentioned in the above section, server pausing may have significant implications on your plugin's functionality. Please make sure to test your plugins on a paused server or to otherwise warn users against enabling the
feature. If you are sure your plugin does not and cannot support server pausing, please use Server#allowPausing(Plugin, Boolean) to prevent accidental use of the feature by users. Similarly you can also check whether it is currently enabled by calling Server#isPaused. We have finally merged API to add or edit (almost) all data components on items.
Since 1.20.5, item data is no longer held in mostly arbitrary NBT, but in properly defined data structures, which have also seen a massive amount of new features that the current ItemMeta API is either missing or poorly representing. You can see the various data types under DataComponentTypes, although we will keep adding getter/setter helper
methods to ItemStack or ItemMeta where appropriate. Here is an example: Java: ItemStack itemStack itemStack (Material.DIAMOND HELMET); // Update parts of the already existing equippable data: // Use the netherrite helmet model when worn and change the equip sound Equippable. Builder equippable =
itemStack.getData(DataComponentTypes.EQUIPPABLE).toBuilder() .model(Material.NETHERITE_HELMET.getDefaultData(DataComponentTypes.EQUIPPABLE).model()) .equipSound(SoundEventKeys.ENTITY_GHAST_HURT); itemStack.setData(DataComponentTypes.EQUIPPABLE).model()) .equipSound(SoundEventKeys.ENTITY_GHAST_HURT).equipSound(SoundEventKeys.ENTITY_GHAST_HURT).equipSound(SoundEventKeys.ENTITY_GHAST_HURT).equipSound(SoundEventKeys.ENTITY_HURT).equipSound(SoundEventKeys.ENTITY_HURT).equ
FoodProperties.food() .canAlwaysEat(true) .nutrition(2) .saturation(3.5f); itemStack.setData(DataComponentTypes.FOOD, food); NOTE: This api is marked as @Experimental and follows similar API safety as the registry API. It may change dramatically between Minecraft versions without backwards compatibility attempts. Added
PlayerItemGroupCooldownEvent to listen to cooldowns that may not be directly associated with using an item, since cooldowns are now added via cooldown groups rather than item types. The already existing PlayerItemCooldownEvent be used to relative enum members have
been deprecated. The new members with more appropriate names are: VELOCITY X, VELOCITY X, VELOCITY Y, 
componenents API directly You can now create custom painting art via API and the new RegistryEvents.PAINTING VARIANT. More are coming over time as well - see for more info on how to use them Stable Paper and Velocity 1.21 builds have been released! As always, backups are absolutely mandatory. After upgrading your world to 1.21, you
cannot downgrade back to a lower version! We would like to thank everyone that worked on this update: If you'd like to support PaperMC as a whole, you can find more information at . Thanks to riley and Luck, Paper now bundles spark as its main profiler for diagnosing causes of lag. Timings is now disabled by default and will be entirely removed at
a later date, possibly with 1.22. As a developer, please make sure you remove any custom Timing uses by then. You can see our docs page as well as the GitHub Discussions page for more details and also provide feedback there. The disable-teleportation-check config option is now gone, as this check no longer exists in vanilla. We have also
changed a number of default configuration values to improve the gameplay experience: merge-radius.exp: 3->-1(disabled). vanilla has its own less aggressive merging logic, but generally xp orbs were not a large performance concern; otherwise we might instead offer new options to change vanilla's merge logic merge-radius.item: 2.5->0.5, reflecting
the vanilla default, since the increased value was often seen as disruptive. If you expect large number of the same item types to be lying around close to one another, you can increase this again entity-activation-range.raider: 48->64 Various values under entity-tracking-range have been increased to make sure you can actually see monsters and
players attacking you from farther away (e.g. no longer running into invisible ghasts shooting at you in the nether. Note that this does not affect ticking, only whether they are sent to a player) players: 48->96 monsters: 48->
performance, adding multiple seconds to each individual larger plugin. This has unfortunately been made slightly worse again in 1.21.1 We have already mildly improved on it and are working on reducing it by as much as possible, but in the meantime you can work around it by disabling the cross-version compatibility measures by either optionally
using Paper's plugin loader as a developer, or by using the paper.disableOldApiSupport startup flag. However, the flag will only work if all of your plugins are built against the latest API version. The final Done (7.392s)! For help, type "help" message now shows the time from when the Minecraft server initially bootstrapped. Previously, it was in a kind
```

of weird spot where it only tracked world loading, plugins, and a few other parts of server startup. We have also reinstated vanilla's original Done preparing level message next to the total startup time. As per the last announcement, we now use Mojang mappings at runtime and thus have to remap plugins that might still be using Spigot's mess of

```
mappings. The .paper-remapped folder in the plugin directory caused a bit of confusion, so here's what it does: It stores the remapped plugin jars as well as a cached server jar, so that all of these don't have to be processed during every single server startup. The folder is automatically cleaned up, so you don't need to (and generally shouldn't) touch it
For developers In case you skipped the 1.20.5/6 update, make sure to read its announcement on Mojang mappings use at runtime and our new Brigadier command API. Attribute modifiers no longer have a name and unid and instead make use of a single string key as its identifier. The old constructors and methods are unusable as of now, only the
Paper-added get/removeModifier via uuids have had a temporary compatibility measure put in. If you are using this API, make sure to move to their replacements as soon as possible. Our vanilla chunk gen delegation API already broke with almost every major update and this one is no exceptions. Together with the regenerateChunk method, their
implementation is no longer feasible due to how hard-coupled and stateful a lot of the handling has become. The only way to properly regenerate chunks with structures and everything else attached is to generate a world with the same seeds and to copy over those chunks. The ItemStack class within the API module no longer holds its method
implementations directly, but instead always redirects to a held internal (Craft)ItemStack instance. This won't matter to most people, but it means that you can no longer run unit tests using the API-only ItemStack instance. This won't matter to most people, but it means that you can no longer run unit tests using the API-only ItemStack instance.
a running server instance. We have added ItemMeta#hasDamageValue to check whether the damage item data component has been added to an item. hasDamage will still return whether there is a non-0 amount of damage. For resetting damage, we recommend using resetDamage instead of setting it to 0 to improve item comparison. There's still
other issues thanks to ItemMeta clashing hard with Minecraft's new item data storage, including unfortunate but not really solvable behavioral breaks to item flags. Right now, we're still working on our improved item data component API in the background and will let you know more once it has been merged. End gateway teleportation cancellation
doesn't reset the portal cooldown, so you should use EntityPortalEnterEvent with PortalType.END GATEWAY to check for initial entries Projectile#getWeapon is now nullable If you have previously contributed to our main Paper repository on GitHub and want the email or name that was used on that commit to be changed, you may use the modmail
command on our Discord to tell us the new details. As per our planned repository restructure in the future, the current git history will be overridden, so we thought we might as well fix the details for anyone that needs it. 1 Here are a few examples (results will vary depending on hardware and OS), which will quickly add up the more plugins you have:
ViaVersion takes over two seconds longer to load (from a few hundred ms to 2.5 seconds) WorldEdit takes two and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect takes a second and a half longer (from about 1.5 to 4 seconds) CoreProtect tak
released! As always, backups are absolutely mandatory. After upgrading your world to 1.20.6, you cannot downgrade back to a lower version! The reason for the stable announcement arriving so late is that upstream's item handling has shown to be incredibly broken after the 1.20.5 changes to items. Unfortunately, there still remain a good number of
smaller issues with ItemMeta APIvanilla conversion, but at this point we should have gotten rid of the nastier ones. In any case, please make sure to report any such issues or missing functionality on our issue tracker. We would like to thank everyone that worked on this update (a lot of people and work were needed for a minor update, once again): If
you'd like to support PaperMC as a whole, you can find more information at . Minecraft 1.20.6 requires you to run Java 21. See here on how to update your installed Java version. Mojang has drastically revamped the way itemstacks store their data. While they are still serialized to raw NBT in chunk and entity files, this is no longer true for commands
and storage at runtime. We have created an item command converter page (where you can update your commands etc. If you are a developer, see the below section on how this might affect you. Unlike vanilla/Spigot, Paper will also automatically upgrade commands in command blocks as well as text components in signs when upgrading from an
older server version. If you for whatever reason do not wish to have these upgraded, you can use the Paper. Disable Command Converter system property/startup flag. For this reason, we also highly discourage upgrading your old world with non-Paper servers. Note that the hide-item-meta config option has not yet been updated, but everything else
should work as expected. Our keep-spawn-loaded and keep-spawn-loaded-range config options have been removed, as Mojang has added the spawnChunkRadius gamerule, serving the same function. As announced previously, we have dropped Spigot's mix of partially Mojang mapped, and partially Spigot mapped runtime names
of classes, methods, and fields. On top of that, we have dropped the arbitrary CraftBukkit package relocation version. Plugins compiled against the reobfuscated server will still work via magical plugin remapping that is applied once on startup, as well as reflection rewriting. However, we highly recommend using paperweight-userdev to offer plugin
jars targeting the mapped server, even if just as a secondary jar, as it would greatly benefit the vast majority of your users (well over 80-90%). If you are not using internals and thus run on a Mojang mapped server fine, you should exclude your plugin from being remapped by adding the Mojang-mapped marker to the jar manifest (setting
paperweight-mappings-namespace to mojang). Alternatively, you can add the entry manually in your gradle or maven build scripts. We have finally added more powerful API to interact with Brigadier commands directly. Brigadier is the command library vanilla uses for creating and parsing commands, meaning you can add much nicer auto-
completions and argument handling for your commands, although this will be most interesting for command libraries wrapping around Brigadier. Basic usage is explained on our docs page. If you were using API to modify item data, then you are largely fine. Since we have long deemed moving away from raw NBT data storage inevitable, we have
never added API for interacting with the underlying NBT data directly. As such, only plugins unnecessarily mutating items through internals should break. The persistent data container API is also not affected. For item serialization, we highly recommend using our ItemStack#serializeAsBytes method over Spigot's config serialization, so that you can
guarantee proper upgrade paths, compatibility for stored items, and better performance. ItemMeta is very much not compatible with the idea of storing any data on any item, e.g. adding durability to a book, or making stone eatable. While we have tried our best at making sure that such custom data is not lost, the API has no proper way of applying
these at the moment. On top of that, a lot of Spigot API that previously assumed a specific set of hardcoded enums, such as banner patterns, break when trying to add custom ones. We currently have a better system for this in the works, but it will take some more time before it is fully ready. Finally, a lot of item data now has hard limits for size and
length, such as lore being limited to 256 entries - most of these you shouldn't really run into. The most notable example is player profiles (including player head profiles), where player names can no longer exceed 16 characters and are limited to printable ascii characters. We strictly enforce these everywhere, as otherwise it could lead to chunks and
player data unable to be saved, as well as inventory or player info updates erroring on the client. Another piece of broken API is our canPlaceOn and canBreak methods, as these have had major changes that can now longer accurately be represented by just a list of materials. New API for this will also be added soon. Unlike before, you can no longer
set most of the hide item flags without the data in question, e.g. you need to set can_place_on data in order to hide it. Finally, here is a closing note from our master of words, electroniccat: as always, BACKUP YO SHIT, so long, and thanks for the fish As many of you might have noticed, Waterfall hasn't received much love from our team and the great
contributor community in the past years. We have also seen less and less traffic in the support channels on Discord. Additionally, Mojang is making huge investments into the game which results in big and complicated changes to the inner workings of the game. While these changes are very welcome and we have been pushing for
some of them for years, they also mean that there is a bunch of work ahead of us for adapting our projects to these changes. We don't think we can find enough people from our flagship projects Paper and Velocity. We also don't feel comfortable putting
out something that doesn't live up to our standards in terms of the testing that went into it. That's why we decided that we want to officially announce the end of life of Waterfall. Starting today, big red angry banners will appear on the Waterfall sub-pages of our documentation site and our website. These are pointing here and act as a way to inform
everybody of what is going on. Other than that, there will be no direct change is that you will still be accessible, you will still be accessible, you will still be able to download all versions of Waterfall as usual. What will change is that you will see even more sporadic updates. You also shouldn't count on updates to new Minecraft versions, although we aren't ruling that out
at this time. Migrate to Velocity! All the knowledge the people who originally worked on Waterfall gained has been put into Velocity, a proxy solution that was built from the ground up with performance, stability and security in mind. You can learn how to get started with Velocity on our documentation site. You can find plugins compatible with
Velocity on Hangar, our new plugin repository. If you encounter any issues while migrating to Velocity, feel free to post on the forums or our Discord, we are happy to help! Please join our discord community if you have any concerns about this announcement. If you are not a developer but a server owner, this might still be important for you. Check
the bottom section on what action you might have to take. Update: This change has been put into effect in 1.20.5, make sure you test with the latest Paper 1.20.5 builds. As already announced before, at some point in the foreseeable future, we will remove the CraftBukkit package relocation (e.g. v1 20 R3). This may be as soon as 1.20.5, as we expect
almost every plugin using internals to break due to major changes in vanilla anyways. Note: This also includes testing of automated remapping of plugins to make them run on Mojang mapped servers, even if a plugin is compiled against the obfuscated class and method names (if you don't use any vanilla internals, this doesn't affect you). So even if
you have already fixed CB package parsing, please check whether your plugins are able to run on this jar if they are using internals (or "nms"). Once these changes are present in stable Paper builds, you can expect a much better experience when using the often dreaded internals by using our userdev Gradle plugin. Most notably, small Minecraft
updates will no longer unconditionally break your plugins. If you reflect on CB classes Easy, just don't try to parse the package version. The following will work on servers with and without CB relocation: Java: private static final String CRAFTBUKKIT PACKAGE = Bukkit.getServer().getClass().getPackage().getName(); public static String
cbClass(String className) { return CRAFTBUKKIT PACKAGE + "." + className); } Class.forName(cbClass("entity.CraftBee")) If you try to parse the server version Do NOT do this: Java: String craftBukkitPackage = Bukkit.getServer().getClass().getPackage().getName(); // This is the *bad* part, including any other parsing of the version String
version = craftBukkitPackage.substring(craftBukkitPackage.lastIndexOf('.') + 1); if (version.equals("v1 20 R3")) { // ... } else { // Unsupported } Instead, use long-standing API: Java: // Paper method that was added in 2020 // Example value: 1.20.4 String minecraftVersion = Bukkit.getServer().getMinecraftVersion(); // Bukkit method that was added in 2020 // Example value: 1.20.4 String minecraftVersion = Bukkit.getServer().getMinecraftVersion(); // Bukkit method that was added in 2020 // Example value: 1.20.4 String minecraftVersion = Bukkit.getServer().getMinecraftVersion(); // Bukkit method that was added in 2020 // Example value: 1.20.4 String minecraftVersion = Bukkit.getServer().getMinecraftVersion(); // Bukkit method that was added in 2020 // Example value: 1.20.4 String minecraftVersion = Bukkit.getServer().getMinecraftVersion(); // Bukkit method that was added in 2020 // Example value: 1.20.4 String minecraftVersion = Bukkit.getServer().getMinecraftVersion(); // Bukkit method that was added in 2020 // Example value: 1.20.4 String minecraftVersion = Bukkit.getServer().getMinecraftVersion(); // Bukkit method that was added in 2020 // Example value: 1.20.4 String minecraftVersion = Bukkit.getServer().getMinecraftVersion(); // Bukkit method that was added in 2020 // Example value: 1.20.4 String minecraftVersion = Bukkit.getServer().getMinecraftVersion(); // Bukkit method that was added in 2020 // Example value = Bukkit.getServer().getMinecraftVersion(); // Bukkit method that was added in 2020 // Example value = Bukkit.getServer().getMinecraftVersion(); // Bukkit method that was added in 2020 // Example value = Bukkit.getServer().getMinecraftVersion(); // Bukkit method that was added in 2020 // Example value = Bukkit.getServer().getMinecraftVersion(); // Bukkit method that was added in 2020 // Example value = Bukkit.getServer().getMinecraftVersion(); // Bukkit method that was added in 2020 // Example value = Bukkit.getServer().getMinecraftVersion(); // Bukkit method that was added in 2020 // Example value =
2011 // Example value: 1.20.4-R0.1-SNAPSHOT String bukkitVersion = Bukkit.getServer().getBukkitVersion(); if (minecraftVersion.equals("1.20.4")) { // ... } else { // Assume latest package version is valid by catching // ClassNotFoundException with:
Class.forName("org.bukkit.craftbukkit.v1 20 R3.CraftServer") } The Minecraft version strings you can parse and evaluate to your heart's content. Another (less recommended) alternative is getting the server protocol version from Bukkit.getUnsafe. If you are running a somewhat recent server version with up-to-date plugins (!), you should also test
whether your plugins are able to run on the test server jar also linked above. Please make sure not to use the server jar on your main servers, but to copy your plugin setup to a separate test server jar on your main servers, but to copy your plugin setup to a separate test server. The error that plugin developers need to fix will look something like this: [11:46:19] [Server thread/ERROR]: Error occurred while enabling
PLUGINNAME v1 (Is it up to date?) java.lang.ArrayIndexOutOfBoundsException: Index 3 out of bounds for length 3 at ... Alternatively, they might log an error saying you are using an unknown or unsupported server version. If any of your plugins start printing new errors like these and you have made sure that they are already up-to-date, please
report the error with a link to this announcement to the relevant plugin authors. Wishing you all a super happy New Year! It's been a big year at Paper! We've grown a lot and have some big changes on the horizon. Our team has yet again increased in size and become even more motivated to work towards our goal of hard forking. PaperMC is
powered by the contributions from everyone, and we have made it our priority that new contributions are getting out there as soon as possible. And through that, we have seen some new faces pop up and contribute more often. This year, we merged over 475 Pull Reguests from over 120 unique contributors! Going through each one of these PRs
wouldn't be possible without you, so we are so grateful for all the bugs reported, testing done, and all the new contributors who decided to give it a shot. This year was a big team effort, and we want to thank each and every person who's been a part of it. Your ideas and hard work made PaperMC even better, and we're so excited to keep growing and
improving together. The future is bright, we have a lot of work being done behind the scenes that we hope to be able to get out into your hands in 2024. With our new Paper Plugins introduced ways of running code much earlier before the server has started. Using this new Lifecycle API, we will now allow plugins to start
running code on an event-based system much earlier in server initialization as well. Java: @Override public void onEnable() { final LifecycleEvents.DUMMY_STATIC.newHandler(event -> { final DummyResourceRegistrar registrar = event.registrar()}
System.out.println("dummy static hook FIRST"); }).priority(-1)); lifecycles.registerEventHandler(LifecycleEvents.DUMMY STATIC.newHandler(event -> { final DummyResourceRegistrar registerEventHandler(event.println("dummy static hook FOURTH (monitor)"); }).monitor());
lifecycles.registerEventHandler(LifecycleEvents.DUMMY STATIC.newHandler(event -> { final DummyResourceRegistrar registrar regi
system through brigadier. This will allow these commands to be usable in things like datapack functions. This API will also support adding custom serverside arguments and more, allowing a more powerful approach compared to the current Bukkit command API. Java: CommandBuilder.of(plugin, "admin") .then(
LiteralArgumentBuilder.literal("execute") .redirect(Bukkit.getCommandDispatcher().getRoot().getChild("execute")) .then( LiteralArgumentBuilder.literal("execute") .redirect(Bukkit.getCommandDispatcher().getRoot().getChild("execute")) .then( LiteralArgumentBuilder.literal("signed_message").then( RequiredArgumentBuilder.literal("execute")) .then( LiteralArgumentBuilder.literal("execute")) .executes().executes().executes().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().getRoot().get
getArgument("msg", MessageArgumentResponse.class); // Gets the raw argument. // This is a better way of getting signed messages, // includes the concept of "disguised" messages, // inc
.getBukkitSender() .sendMessage(signedMsg, ChatType.SAY COMMAND.bind(comp)); }); return 1; }) )) .description("Cool command showcasing what you can do!") .aliases("alias for admin that you shouldnt use", "a") .register(); See more information here We've recently introduced autogenerated API keys in our API, and in general are closing
the gap allowing us to properly implement custom type registration. In this API, we finally allow custom types to be registered vanilla types. Java: static final TypedKey NEW_EVENT = GameEventKeys.create(Key.key("machine_maker",
"best event")); @Override public void bootstrap(@NotNull BootstrapContext context) { final LifecycleEventManager lifecycles.registerEventHandler(RegistryEvents.GAME EVENT.preFreeze().newHandler(event -> { // the
RegistryView provided here is writable so you can register (NEW EVENT, builder.range(2); }); })); // registers a handler for the addition event lifecycles.registerEventHandler(RegistryEvents.GAME EVENT.newAdditionHandler(event -> { // checks if the object being registered is the block open games.
event if (event.key().equals(GameEventKeys.BLOCK_OPEN)) { // multiplies the range by 2 event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(event.builder().range(e
we have been able to create a stable way of deobfuscating the game in a way more friendly than before. More work than ever is being put into our tech for hard forking, and we are excited to be able to work off of some of the best tech developed for deobfuscation. Stable Paper and Velocity 1.20.4 builds have been released! As always, backups are
absolutely mandatory. After upgrading your world to 1.20.4, you cannot downgrade back to a lower version! We would like to support PaperMC as a whole, you can find more information at . From now on, instead of creating a
new Discord channel for every update, we will post important milestone updates (such as the availability of experimental builds) into the new update-announcements channel sto your list via "Channels & Roles" at the top of the channel list first.
With the new sendResourcePacks and removeResourcePacks and removeResourcePacks methods, you can give each pack its own UUID to be individually added and removed later, which means that you can have multiple packs applied at once! The existing setResourcePack method will override all previous ones to retain expected behavior. Keyed provides a NamespacedKey
getKey() to get keys for biomes, item and block types, sounds, etc. However, trim patterns and trim materials mark the first two registry based objects that do not require a key in all cases, please use the newly added Registry#getKey(Object).
While the getKey methods will be available until actually broken, using the method on Registry will make sure your plugin does not suddenly break later. Note that because of the possibility of no key existing, this method is nullable. If you are sure one will exist, you can also use the nonnull Registry #getKeyOrThrow. As per the last big announcement
we now have our own website for you to upload your Paper, Bungee, and Velocity plugins to: If you don't feel like manually uploading your builds to it, you can also check out our hangar users: You can now use your GitHub, Google or
Microsoft account to login to Hangar. If you don't have an account you can signup using one of these OAuth providers on the security settings. Note that this functionality, while thoroughly tested, is still a bit experimental and the UX of the flows and
the design of the UI is still subject to change. Please send us your feedback on Discord or via the issue tracker. Stable Paper and Velocity 1.20.2 builds have been released! As always, backups are absolutely mandatory. After upgrading your world to 1.20.2, you cannot downgrade back to a lower version! We would like to thank everyone that worked
on this update (a lot of people and work needed for a minor update!): If you'd like to support PaperMC as a whole, you can find more information at . Due to larger network changes and perfectly timed holidays of a few of our devs, it took a little longer to get Velocity ready for 1.20.2. Plugins manually sending packets will need updating. The most
notable change in user behavior here is that on server switches, the Minecraft client will now drop its current resource pack, meaning it will have to be re-sent if you want to keep it across backend servers. Velocity will re-apply the pack you set via Velocity API, but if you send it on the Paper server, you will need to do so on more than just the hub.
This is unavoidable at the moment, but we're hopeful that Mojang is going to address this in a future update. While Waterfall received support was pretty broken for 1.20.2 pretty early on as part of BungeeCord upstream updates, its support was pretty broken for the first few days and weeks after the release and still does not properly handle the new protocol changes
in some places. In general, Waterfall is unlikely to receive our full attention given that Velocity is meant to be its more performant, stable, and secure successor. In similar fashion to us retiring Travertine a while ago, the same will happen to Waterfall in the future. For now though, we will continue providing you with upstream updates at the very
least. This is very important if you for whatever reason use reflection to either parse the relocated package version. call CB internals. At some point in the future, we will only provide jars without relocation, given it is a nonsensical practice resulting in unavoidably bad code design and unexpected incompatibilities in a large number of plugins. While
we will be able to automagically remap both direct and reflective calls to the relocated package, parsing the package version is not supported and WILL break at some point in the future. The changes we have planned should make working with internals a lot easier, since we recognize that sometimes (though not as often as some might think) there is
no better alternative. If you reflect on CB classes Easy, just don't try to parse the package version. The following will work on servers with and without CB relocation: Java: private static final String class().getPackage().getName(); public static String class().fring class().getPackage().getName(); public static String class().getName(); public static String class(); public static String class(); public s
CRAFTBUKKIT_PACKAGE + "." + clazz); } Class.forName(cbClass("entity.CraftBee")) If you try to parse the server version Do NOT do this: Java: String craftBukkitPackage = Bukkit.getServer().getClass().getPackage().getName(); // This is the *bad* part, including any other parsing of the version String version =
craftBukkitPackage.substring(craftBukkitPackage.lastIndexOf('.') + 1); if (version.equals("v1_20_R1")) { // ... } else { // Unsupported } Instead, use long-standing API: Java: // Paper method that was added in 2020 // Example value: 1.20.1 String minecraftVersion = Bukkit.getServer().getMinecraftVersion(); // Bukkit method that was added in 2011 //
Example value: 1.20.1-R0.1-SNAPSHOT String bukkitVersion = Bukkit.getServer().getBukkitVersion(); if (minecraftVersion.equals("1.20.1")) { // ... } else { // Assume latest still works, or error as unsupported // Alternatively for extra compatibility, check if // the latest package version is valid by catching // ClassNotFoundException with
Class.forName("org.bukkit.craftbukkit.v1 20 R1.CraftServer") } The Minecraft version strings you can parse and evaluate to your heart's content. Another (less recommended) alternative is getting the server protocol version from Bukkit.getUnsafe. Enums such as Biome implementing the Keyed interface will be converted to classes with public static
final objects at some point. While some backwards compatibility will be provided, please try to avoid the use of switch statements, EnumMap and EnumSet on these, including the Material enum. With the protocol changes comes the ability to send a resource pack before the player has even joined the world, making previously required precautions
like resource pack servers unnecessary. Finally, it also sends its client settings (including the language) before this, but we will let you know once it is added! As per the last big announcement, we now have our own website for you to upload your Paper,
Bungee, and Velocity plugins to: If you don't feel like manually uploading your builds for Paper 1.20 have been released. As always, backups are absolutely mandatory. After upgrading your world to 1.20, you cannot downgrade back to a lower
version! We would like to thank everyone that worked on this update: Next to those people, you can find links to support them individually. If you'd like to support them individually indi
SmithingTrimRecipe and SmithingTransformRecipe InventoryType. SMITHING now uses 1.20's new smithing table interface Sign#isEditable() has been deprecated in favor of a new method called isWaxed, same for its setter API scheduled for removal in 1.21, so please make sure you remove usages.
of these in your plugins: As per the last big announcement, we now have our own website for you to upload your Paper, Bungee, and Velocity plugins to. There have been little updates in the past few weeks due to us being busy with other projects, including Paper, but rest assured we'll keep working on it! Our downloads API has different channels to
distinguish builds - right now between experimental and default. For future updates, we will no longer provide any early experimental builds on Discord, instead using the experimental and default. For future updates, we will no longer provide any early experimental builds on Discord, instead using the experimental builds on Discord, instead using the experimental builds on Discord, instead using the experimental and potentially
breaking versions. Please adjust your download scripts accordingly. Experimental builds marked as such will be available to download on our homepage as well. Once again, we have another exciting announcement for you, this time about Paper, Velocity, and Waterfall plugins, called Hangar! The
main reason we started working on this is to finally provide a centralized place for Paper and Velocity plugins. Compared to the Spigot forums, Hangar allows you much more control over your resource in terms of: adding other authors to your project, creating organizations with projects under them, managing roles per project or per organization
(such as editor or developer) combined releases with multiple jars or external links per platform, customizable release channels (such as beta or snapshot), a proper, documented API to upload and download plugins (OpenAPI yaml), creating multiple project description/wiki pages, selecting compatibility with minor Minecraft versions as well as
specific Velocity versions, additional tags to mark a plugin as an addon, a library, or Folia compatible, a secure account system with support for modern multi factor authentication standards like TOTP and WebAuthN (YubiKeys are supported!) ... and more. As Hangar is still in beta, we intend on adding a lot more features and quality of life change
such as the ability to set notifications per release channel and creating draft releases. You can follow up on our current plans in the Hangar roadmap project: . There are of course other such resource sites already, such as CurseForge and Modrinth, however, we have made Hangar specifically with Paper and Velocity plugins in mind, so that you can
have the best experience looking for plugins or uploading them to Hangar. We have already made a Gradle plugin you can find here with examples provided: - so the only thing that stands between you and your first uploaded version is creating an account and a project on Hangar!
Additionally, for developers who published to the Spigot forums before, we created an importer for that! You can find it here. It will attempt to import the description and convert it to Markdown, set the project avatar and basic settings such as the category to make it as easy as possible for you to adopt Hangar for your projects. Do note however that
you will have to upload your versions manually after you imported your projects! If you happen to find any bugs, you can report them on our issue tracker: . If you want a testing grounds for the API, please use our staging instance. Let me end this post with a bit of a personal note: Hangar was born out of a discussion on the Papers Discord/IRC manually after you imported your projects! If you happen to find any bugs, you can report them on our issue tracker: . If you want a testing grounds for the API, please use our staging instance.
channel, after being annoyed by existing platforms, almost 3 years ago. First it was a from-scratch project, then we rewrote the backend, then we rewrote the frontend, again. It has been quite a wild ride, with many ups and downs. The lowest
point was December last year: We weren't getting anywhere, I thought everything we had sucked, we were exploring alternatives internally, I started modifying existing software for our needs. Basically, I was ready to give up. In the end, we decided to not go that route, the Paper team assured me that what we had at that time was already better than
 other existing solutions and so I pushed through all that together with Kenny, without whom I would have never be able to do that. So I'll end with thanking him, Machine Maker, AlessioGr, mdcfe and all the other contributors, the Paper team, the people on the Hangar Discord, the plugin developers who where invited as early adopter and everybody
else who tested Hangar over the years and provided valuable feedback, and generally everybody in the Paper community who pinged me daily to remind me that Hangar means something to them: You were annoying as heck, but I am glad you did it. And now everybody go sign up! Paper 1.19.4 and Velocity with 1.19.4 support are now available on
our website! As always, we recommend that you make a backup of your server before upgrading. Remember that you cannot downgrade your Paper server after doing the update. We would like to thank the following people for their work on the update process:
using the experimental channel in our downloads API. This means that you will need to distinguish between channels in your scripts to avoid getting highly experimental builds marked as such will be available to download on our homepage as well. Our
new website is now live at Cubxity has been working on this with us for a while, and we're happy to finally be able to replace our old site that has been an annoyance to maintain for quite some time. Feedback is of course appreciated, the same applies to code contributions: API changes Experimental features have representation in API, but are
marked as experimental and are subject to changes, as Mojang might still change them in major ways before they land in 1.20. See here for more information on experimental features in general. Using adventure's ClickEvent.callback methods, you can now easily register message click event callbacks without having to keep track of them yourself.
This code for example will create a click Event to open a book that can be used for up to 2 minutes and has 5 uses: Java: ClickEvent clickEvent clickEvent clickEvent to open a book that can be used for up to 2 minutes and has 5 uses: Java: ClickEvent clickEvent clickEvent clickEvent clickEvent clickEvent clickEvent and last 5 uses: Java: ClickEvent c
player.sendMessage(Component.text().content("Click me!").clickEvent(clickEvent()); In these methods, you can also make sure only a certain player/players with a certain player/players with a certain permission are allowed to use the callback. LivingEntity#setHurtDirection throws an UnsupportedOperationException if called on a non-player HopperMinecart#setCooldown and
getCooldown throw UnsupportedOperationException WIP registry modification API Machine Maker has been working on API to be able to modify certain Minecraft registries, including damage events. Before merging it, we would like to gather a last round of community feedback on it: Additionally there is a second pull request to add API for
modifying tags, used by the client in a lot of different ways, including knowing which blocks are climbable and which tool you can use to faster dig a block: Future changes regarding API enums such as Biome implementing the Keyed interface will be converted to classes with public static final objects at some point. While some backwards
compatibility will be provided, please try to avoid the use of switch statements, EnumMap and EnumSet on these. Paper 1.19.3 and Velocity with 1.19.3 support are now available on our website! Even though Paper is deemed stable, we still recommend that you make a backup of your server before upgrading. Remember that you cannot downgrade
your Paper server after doing the update. Despite only being a minor version, quite a bit of work has gone in the update process: As explained in a minecraft versions. On servers, they can been being a minor version, quite a bit of work has gone in the update process: As explained in a minecraft versions. On servers, they can been being a minor version, quite a bit of work has gone in the update. We would like to thank the following people for their work on the update process: As explained in a minecraft versions. On servers, they can be update.
enabled by adding update 1 20 to the initial-enabled-packs option in the server properties file (with entries separated worlds. We do not recommend enabling these feature packs on production servers, as the features that come with them (such as Camels and the new bamboo blocks) will not
survive world upgrades and are still riddled with bugs. We do not provide support for these experimental features and will not fix any issues with them, unless the issue in question is caused by one of our patches and can affect other parts of the server as well. Removal of Chat Previews As part of 1.19.3, Mojang have removed the chat preview
functionality in its entirety. This means you cannot make players sign messages that have been changed by the server (unless only formatting of the message if they hover over a modified one. However, this does not affect AsyncChatDecorateEvent and
AsyncChatCommandDecorateEvent; going forward, we will mostly likely encourage changing a message's content through the decorate events, with changes to API regarding chat and signed chat will be held off until 1.20 so it is less likely to break again with Mojang still
doing such major changes to the system. Experimental features Plugin developers can prepare their plugins for these features with prelimary API, but be aware that most of the API representations of the experimental features are likely to change before they are finalized by Mojang. These classes, fields, etc. will be marked with an @Experimental
annotation. We've seen a lot of reports of a new malware going around Minecraft servers. It seems to be spread by compromised Spigot plugin-author accounts, and is somewhat difficult to detect. We do know that the following exception is caused by it: Code: java.net.NoRouteToHostException: No route to host If you see this in your logs, that server
is most likely infected. There are other indicators too - the compromised JAR will have inside of it a file called plugin-config.bin. We do have a one-liner for searching for this in your plugin directories, if you're on a Windows system you can run this command in your plugins directory:
Code: findstr /sml /c:"plugin-config.bin" * Run the above while in your server or plugin directory, and if you get a match, you likely have an infected plugin. If you do not get a match, that is a good thing - you are likely not infected plugin. If you do not get a match, that is a good thing - you are likely not infected plugin. If you do not get a match, that is a good thing - you are likely not infected.
variants of it. We highly suggest users run this tool as it contains checks for a lot more malware sources. If this tool reports any malware found, be sure to double check whether it's a false positive or not (known example: ForceOP check falsely triggers on a handful of plugins because of how it's used in plugins). If you do get a match or think that you
are infected, you should delete all of your JAR files and re-download them, as the malware spreads itself to other JARs. You should also immediately reinstall your machine, as this malware is known to install system services outside of Minecraft. It might be more effort, but it is important that infected machines are reinstalled, or else the malware will
know, it is in a good state for production use - however, it is still early in the release cycle and new bugs may be discovered at any time, so we recommend that you make a backup of your server before upgrading. We have received a great deal of concern regarding our stance and support for the new chat signing aspect of Minecraft. As such, we
figure it best to clear up a few points in this announcement. First: Paper will add support to its API so that plugins, servers, plugins, and developers that wish to disable chat signing will also have the ability to do so without any hindrance
from us. We will not enforce the use of Minecraft's chat signing features nor will we use our software to actively prevent the bypassing of these new safety features. Similarly to offline mode servers, we would not at the same time
prevent people's actually doing it, and will leave that decision up to each server owner and anyone using our software. Fourth: Paper is not so involved with the community for years in multiple ways, but we have done all that we can do by
 bringing up various concerns about the new system with them and leaving it in their hands. Finally: There is more information is made available. If you'd like to help the project's infrastructure costs, feel free to check out - we greatly
appreciate your continued support! Also, a special thanks to kennytv and Machine Maker for their extraordinary help on this update. They both have GitHub Sponsors if you'd like to contribute directly to them! We recently announced on our discord that a new chunk system is in development and needs testing. It has been updated to include 1.19.1
You can find more information at the PR on our GitHub. Please do not test this on worlds that are not backed up, or on production servers, as it is unstable and will probably break things. We appreciate any help in identifying and reporting bugs - please report anything broken on the same PR, linked again here. One more reminder to please back up
your server before upgrading to 1.19.1 - it is very important to do this! We're happy to announce that initial builds for Paper 1.19 have been released. We were able to fix a lot of issues already, but there might still be breaking ones, so as always, backups are absolutely mandatory. After upgrading your world to 1.19, you cannot downgrade back to a
lower version! We would like to thank everyone that worked on this update. Not only the people actually working on the code, but everyone that provided feedback, helped us test, has been patient, and the people that joined us in VC or on Twitch (where over 400 people watched Kenny stream at one point!). We would like to especially thank the
following people:Next to those people, you can find links to support them individually. If you'd like to support PaperMC as a whole, you can find more information at . Starting with 1.19, the Paper configuration files will now be split into multiple parts. Instead of having one giant paper.yml file for everything, there are now two files in a newly created
config directory: paper-global.yml, where you can configure options that apply to the whole server, and paper-world-defaults.yml, where you can set default per-world values; you can configure options that apply to the whole server, and paper-world-defaults.yml, where you can set default per-world values; you can set default per-world values; you can configure options that apply to the whole server, and paper-world-defaults.yml, where you can set default per-world values; you can configure options that apply to the whole server, and paper-world-defaults.yml, where you can set default per-world values; you can configure options that apply to the whole server, and paper-world-defaults.yml, where you can set default per-world values; you can configure options that apply to the whole server, and paper-world values; you can set default per-world values; you can 
into each individual world directory (paper-world.yml), so for example, for the world the end, you will find the configuration file at world the end/paper-world.yml. In addition to this, all configuration file at world the end/paper-world.yml. In addition to this, all configuration file at world the end, you will find the configuration file at world the end, you will find the configuration file at world the end, you will find the configuration file at world the end, you will find the configuration file at world the end, you will find the configuration file at world the end, you will find the configuration file at world the end, you will find the configuration file at world the end, you will find the configuration file at world the end, you will find the configuration file at world the end, you will find the configuration file at world the end, you will find the configuration file at world the end, you will find the configuration file at world the end, you will find the configuration file at world the end, you will find the configuration file at world the end, you will find the configuration file at world the end, you will find the configuration file at world the end, you will find the configuration file at world the end, you will find the end of the end, you will find the end of the en
in the paper configs. Instead, you use MiniMessage, which allows modern formatting with RGB colors, gradients, translatable components, and a lot more. You can find more information about MiniMessage here: To try out MiniMessage formatting, you can use this live-previewing website: . Your current configuration files will be migrated into the
new format automatically while keeping all of your previous settings. Your old paper.yml old. Our documentation will be updated over the next couple of weeks to reflect those changes. Alternate Current redstone implementation About a month ago, Space Walker ported his Fabric
mod to Paper, allowing us to offer another redstone implementation: Alternate Current. You can enable it by changing the per-world setting redstone-implementation to alternate current. As of now, Alternate Current is faster and more stable than the already implementation to alternate current. You can enable it by changing the per-world setting redstone-implementation to alternate current. As of now, Alternate Current is faster and more stable than the already implementation to alternate current.
slightly deviates from Vanilla in certain edge cases, such as the order of surrounding block updates. Read more about Alternate Current and how it differs from other redstone implementations on its README. Signed chat messages Minecraft 1.19 introduced client-side signing of chat messages, allowing other clients to verify a message has been sent
messages will be sent as (unsigned and unverified) system messages - this has no meaningful impact on how clients display these messages yet. With Mojang trying to make the player chat more secure, we will have to make some larger additions and changes around message events and API in the future to allow features like the ability to preview
formatted messages on the client. MiniMessage (String) and sendRichMessage (String) methods in API After the inclusion of MiniMessage text formatting - we
strongly discourage the use of the old sendMessage(String) methods using legacy formatting. Configurate We are currently not exposed once Configurate receives more updates to make it more user friendly for use in plugins. Our downloads API has
different channels to distinguish builds - right now between experimental and default. The first few 1.19 builds were released in the experimental channel in our
downloads API. This means that you will need to distinguish between channels in your scripts to avoid getting highly experimental builds marked as such will be available to download on our homepage as well. We are now releasing initial builds for
1.18.2. These have been tested by our team over the last few days and we were able to iron out quite a few issues, but you should still be careful. These are early builds, they may contain breaking issues, backups are absolutely mandatory! After you update, you cannot downgrade your world back to 1.18.1 or lower again! As always, we would like to
thank everybody who contributed to this update, be it by contributing code, reporting issues or just discussing changes with us in voice chat and cheering us on. In particular, we would like to thank the following developers: You can also support the PaperMC project itself, you can find more information about that here: MiniMessage included in Paper
API Since MiniMessage is now stable, we have decided to include it in the Paper API. For those who don't know what MiniMessage is yet; it's a simple, user-friendly, string representation of Minecraft's chat components, perfect for use in config files. You can learn more about it here, and be sure to check out the Web UI to start playing with
MiniMessage now! Note: While MiniMessage is now packaged with the API and available for plugins to use, it is not currently used by Paper itself. In the future, we plan to migrate all configurable messages over to MiniMessage, allowing for much more control over the styling of configurable messages! StructureLocateEvent replacement Due to a lot
of internal restructuring in Vanilla, the StructureLocateEvent has been replaced by the StructureStructures (as opposed to just one StructureType previously). The old event is no longer fired, so if you used it, you should update the API usage as soon as possible. Another thing we would like to note is that we
are working on a new documentation site for the PaperMC organization, which will replace the various documentation sites currently in use. We are looking to open the floor for discussion, ideas, suggestions, and contributions, so please keep an eye on the #docs channel for more information! After the initial release a bit over a month ago, Paper
1.18.1 is now deemed ready for use in production environments. As with any upgrade, please ensure you have a functioning backup before proceeding. World downgrades are not supported under any circumstances. Upgrading worlds using the
old (0 to 256) height to upgrade cleanly to the new (-64 to 320) height. Retrogen will fill in new caves below the bedrock layer, allowing you to explore the new terrain in already generated chunks. Should retrogen be undesired, you can disable it by setting below-zero-generation-in-existing-chunks to false in spigot.yml. This option is not recommended
and may not work correctly in conjunction with --forceUpgrade or with worlds older than 1.14. Mojang has also introduced world blending to cleanly transition from old to new generation Ores can now generate a lot higher than before, so you might
need to adjust your Anti-Xray settings. We have increased the default max-block-height to 64, but you might want to increase it even further. Please be aware that higher numbers might impact performance, especially with engine-mode: 2. See stonar's anti-xray guide and the updated ore distribution for more information. Security fixes to all
Minecraft versions Throughout December, we have pushed important security fixes to all Paper versions, you should update immediately. While we decided to make an exception in pushing the fixes to legacy versions, we will never do this again, and it is
only a matter of time until a new major security issue sees the light of day. We would also like to thank the member of our community that made us aware of the issue before it received too much public attention and to have Minecraft as a whole be one of the
first communities to warn users about it. Contributors to 1.18 A big thank you to everyone who contributed to the update process: If you want to support the PaperMC project, you can find more information here: . We would also like to thank everyone who watched, chatted and talked with us during the update process. You are amazing Changes to
Paperweight (Paper contributors) Instead of using the shadowJar and reobfJar Gradle tasks to create a runnable (but not distributable) jars are now created with the createMojmapPaperclipJar or createReobfPaperclipJar tasks. Similarly, Paperclip (distributable) jars are now created with the createMojmapPaperclipJar or createReobfPaperclipJar tasks. Similarly, Paperclip (distributable) jars are now created with the createMojmapPaperclipJar or createReobfPaperclipJar tasks. Similarly, Paperclip (distributable) jars are now createMojmapPaperclipJar or createReobfPaperclipJar tasks.
get a full list of tasks by running gradlew tasks. An updated, in-depth guide on contributing to Paper can be found here. Paperweight Userdev: Working with NMS in 1.17+ (Plugin developers using NMS) After upstream dropped their field mappings in 1.17, the same now happened with method names as well. Even though we generally advise against
depending on server internals, we understand that not everything is possible through API. As of now, the only feasible way of depending on server internals is by coding against mapped names, which are then compiled to the obfuscated names to run with the obfuscated server. Paperweight's Userdev allows you to do exactly that, but unlike
upstream's maven plugin, userdev uses full Mojang mappings with additional yarn parameter mappings, so you can more easily update your plugin whenever a new Minecraft version is released. Userdev is the only supported way of working with NMS in 1.18+. The obfuscated jar is no longer valid to compile against. To set up paperweight userdev: A
full, working example can be found on GitHub. Both settings.gradle.kts and build.gradle.kts are important! Paperweight Userdev integrates with the Gradle supports automatic migration for the majority of project configurations. We would
recommend the Kotlin DSL, which can be selected via gradle init --dsl kotlin. If you have any issues getting started with Userdev, please come by the #paper-dev channel on our Discord. Downloads API when trying to
download Paper via scripts and automated tools. For reference, see the Downloads API documentation. We have also introduced a channel field to the paperMC team. Larry has
been promoted to the role of Community Manager and will be focusing on the moderation aspects of the community. We would also like to welcome ocelotpotpie to our moderation team. sulu has taken the position of Triage lead and will be responsible for managing the Triage team, which looks after our GitHub issues. jmp has joined the Maintainer
team; Proximyst has left the team, and we wish her all the best! aurora has left the development team to focus more on her responsibilities as a community - here, we ask questions, give answers, and talk about everything to do with our
 projects. Paper: This is our Minecraft server software. It's designed to be fast, bug-free, and a breeze to use. With an extensive API for plugin developers to boot, we work hard on Paper and we're proud of it. Velocity: A recent addition to the PaperMC community, Velocity is the most modern, secure, and highly
                                                                                                                                                                                                                                                                                                                                                                                                                   performant proxy for Minecraft servers
out there. Waterfall: A BungeeCord-compatible Minecraft server proxy that might not be quite as fast as Velocity, but is fully supported and fully supported and fully supported and fully supported and fully support any BungeeCord plugins you might need to use! Overall, PaperMC is a community that's excited about Minecraft software and making it better, while also providing a community space that's fun to
be in. We have a strong culture of helping people and sharing knowledge for the betterment of everyone involved. Regardless of if you're a developer, run a Minecraft server yourself, or help someone else do it, we hope that you find yourself welcomed (and welcome others!) whenever you join us. Whether here, on Discord, or on GitHub, enjoy your
time in the PaperMC community! If you care even one atom about stability, no, a server with both plugins and mods is not possible. They are 2 very different systems that work in very different ways. In addition, many of them do sketchy things to attempt to fix their stability issues, i.e. here's a post from the EssentialsX team detailing why not to use
Mohist: I'm not going to bother responding to that response from Mohist, since it seems it's mostly pushing the blame onto EssX, and I'm guessing staff would yell at me. The long and short of it is, Mohist is unstable, and has done sketchy things in the past, meaning they're likely to again. Use it at your own risk, and remember to take backups. I'm not
going to bother responding to that response from Mohist, since it seems it's mostly pushing the blame onto EssX, and I'm quessing staff would vell at me. The long and short of it is, Mohist is unstable, and has done sketchy things in the past, meaning they're likely to again. Use it at your own risk, and remember to take backups. Yea, Mohist is a very
wierd software, by me (on a test server) it works, but without knowing of it's savety Paper and Velocity 1.21.3 builds are out of the experimental phase! As always, backups are absolutely mandatory. After upgrading your world to 1.21.3, you cannot downgrade back to a lower version! We would like to thank everyone that worked on this update:If
you'd like to support PaperMC as a whole, you can find more information at . Timings removalAfter having added spark as our main profiler for diagnosing causes of lag in 1.21, Timings has been set to no-op mode. This means that it can no longer be enabled or used, though its API classes will remain until a later update. As a developer, please make
sure you remove any custom Timing uses by then. You can see our docs page as well as the GitHub Discussions page for more details and also provide feedback there. Server pausing when no players are online after a while. This behavior is
disabled by default on Paper because it is incompatible with what plugins expect and might do with no players online. You can enable it again by changing the value in the server pausing or you may run into crashes or
save data issues (they won't be able to properly work with entities and the world, or do other actions that would require "active" world ticking). Unlike the Bukkit schedulers, Folia's GlobalRegionScheduler will not be ticked while the server is paused. If you ran a Spigot 1.21.3 server before switching back to Paper, we recommend manually setting
pause-when-empty-seconds to -1 to disable it. Configurable entity despawn time under entities from being kept alive permanently. This patch was ported from pause-when-empty-seconds to -1 to disable it. Configurable entity despawn times in ticks for when an entity should be forcefully despawned. An example usecase of this is preventing certain projectiles from being kept alive permanently. This patch was ported from
Pufferfish with Kevin's go-ahead. Option to use old enderpearl behavior config option to prevent ender pearls from being saved to the player and loading chunks, meaning they will behave like they did in 1.21.1 and before. Paper will default to the new vanilla behavior. Invulnerability damage reduction
We have fixed incorrect handling of damage reduction during invulnerable ticks after being hit to work like it does in Vanilla again. String duplication fixed by MojangMojang fixed string disarming behavior and its dupe, so we have dropped our patch and configuration option. For developers In case you skipped the 1.20.5/6 update, make sure to read
its announcement on Mojang mappings use at runtime and our new Brigadier command API. Server pausing may have significant implications on your plugin's functionality. Please make sure to test your plugins on a paused server or to otherwise warn users against enabling the feature. If you are
sure your plugin does not and cannot support server pausing, please use Server#allowPausing(Plugin, Boolean) to prevent accidental use of the feature by users. Similarly you can also check whether it is currently enabled by calling Server#isPaused. Extensive Item DataComponent APIWe have finally merged API to add or edit (almost) all data
components on items. Since 1.20.5, item data is no longer held in mostly arbitrary NBT, but in properly defined data structures, which have also seen a massive amount of new features that the current ItemMeta API is either missing or poorly representing. You can see the various data types under DataComponentTypes, although we will keep adding
getter/setter helper methods to ItemStack or ItemStack or ItemStack or ItemStack or ItemStack itemStack or It
itemStack.getData(DataComponentTypes.EQUIPPABLE).toBuilder() .model(Material.NETHERITE HELMET.getDefaultData(DataComponentTypes.EQUIPPABLE).model()) .equipSound(SoundEventKeys.ENTITY GHAST HURT); itemStack.setData(DataComponentTypes.EQUIPPABLE).model()) .equipSound(SoundEventKeys.ENTITY GHAST HURT).model() .equipSound(SoundEventKeys.ENTITY GHAST HURT).model() .equipSound(SoundEventKeys.ENTITY GHAST HURT).model() .equipSound(SoundEventKeys.ENTITY HURT).model() .equipS
FoodProperties.food() .canAlwaysEat(true) .nutrition(2) .saturation(3.5f); itemStack.setData(DataComponentTypes.FOOD, food); NOTE: This api is marked as @Experimental and follows similar API safety as the registry API. It may change dramatically between Minecraft versions without backwards compatibility attempts. Other API changes Added
PlayerItemGroupCooldownEvent to listen to cooldowns that may not be directly associated with using an item, since cooldowns are now added via cooldown groups rather than item types. The already existing PlayerItemCooldownEvent to listen to cooldown groups rather than item types. The already existing PlayerItemCooldownEvent to listen to cooldown groups rather than item types.
been deprecated. The new members with more appropriate names are: VELOCITY X, VELOCITY Y, 
componenents API directly You can now create custom painting art via API and the new Registry Events. PAINTING VARIANT. More are coming over time as well - see for more info on how to use them Last edited by a moderator: Dec 6, 2024 Messages 27 Reaction score 150 Points 3 Page 2 Paper and Velocity 1.21.3 builds are out of the experimental
phase! As always, backups are absolutely mandatory. After upgrading your world to 1.21.3, you cannot downgrade back to a lower version! We would like to support PaperMC as a whole, you can find more information at . Timings removalAfter having added spark as our main profiler for
diagnosing causes of lag in 1.21, Timings has been set to no-op mode. This means that it can no longer be enabled or used, though its API classes will remain until a later update. As a developer, please make sure you remove any custom Timing uses by then. You can see our docs page as well as the GitHub Discussions page for more details and also
provide feedback there. Server pausing when empty (disabled by default) Vanilla added a server properties option to pause world and entity ticking when no players are online after a while. This behavior is disabled by default on Paper because it is incompatible with what plugins expect and might do with no players online. You can enable it again by
changing the value in the server properties file, but we generally recommend against doing that unless you are 100% certain your plugins are compatible with server pausing or you may run into crashes or save data issues (they won't be able to properly work with entities and the world, or do other actions that would require "active" world ticking).
Unlike the Bukkit schedulers, Folia's GlobalRegionScheduler will not be ticked while the server is paused. If you ran a Spigot 1.21.3 server before switching back to Paper, we recommend manually setting pause-when-empty-seconds to -1 to disable it. Configurable entity despawn timeUnder entities.spawning.despawn-time, you can now configure
hard despawn times in ticks for when an entity should be forcefully despawned. An example usecase of this is preventing certain projectiles from being kept alive permanently. This patch was ported from Pufferfish with Keyin's go-ahead. Option to use old enderpearl behavior.
ender pearls from being saved to the player and loading chunks, meaning they will behave like they did in 1.21.1 and before. Paper will default to the new vanilla behavior. Invulnerabilet to keep they did in 1.21.1 and before they did in 1.21.1 and before. Paper will default to the new vanilla behavior. Invulnerability damage reduction.
duplication fixed by MojangMojang fixed string disarming behavior and its dupe, so we have dropped our patch and configuration option. For developers In case you skipped the 1.20.5/6 update, make sure to read its announcement on Mojang mappings use at runtime and our new Brigadier command API. Server pausingAs mentioned in the above
section, server pausing may have significant implications on your plugin's functionality. Please make sure to test your plugin does not and cannot support server pausing, please use Server#allowPausing(Plugin, Boolean) to prevent accidental use
of the feature by users. Similarly you can also check whether it is currently enabled by calling Server#isPaused. Extensive Item DataComponent APIWe have finally merged API to add or edit (almost) all data components on items. Since 1.20.5, item data is no longer held in mostly arbitrary NBT, but in properly defined data structures, which have also
seen a massive amount of new features that the current ItemMeta API is either missing or poorly representing. You can see the various data types under DataComponentTypes, although we will keep adding getter/setter helper methods to ItemStack or ItemMeta where appropriate. Here is an example: ItemStack itemStack = new
ItemStack(Material.DIAMOND HELMET); // Update parts of the already existing equippable data: // Use the netherrite helmet model when worn and change the equippable = itemStack.getData(DataComponentTypes.EQUIPPABLE).toBuilder()
.model(Material.NETHERITE HELMET.getDefaultData(DataComponentTypes.EQUIPPABLE).model()) .equipSound(SoundEventKeys.ENTITY GHAST HURT); itemStack.setData(DataComponentTypes.EQUIPPABLE).model() .equipSound(SoundEventKeys.ENTITY GHAST HURT).model() .equipSound(SoundEventKeys.ENTITY HURT).model() .equipSound(SoundEventKeys.ENTITY HURT).model() .equipSound(SoundEventKe
.saturation(3.5f); itemStack.setData(DataComponentTypes.FOOD, food); NOTE: This api is marked as @Experimental and follows similar API safety as the registry API. It may change dramatically between Minecraft versions without backwards compatibility attempts. Other API changes Added PlayerItemGroupCooldownEvent to listen to cooldowns
that may not be directly associated with using an item, since cooldown groups rather than item types. The already existing PlayerItemCooldownEvent extends the new event. Due to Vanilla changes to relative teleportation, TeleportFlag, Relative enum members have been deprecated. The new members with more
appropriate names are: VELOCITY X, VELOCITY Y, VELOCITY Y, VELOCITY Z, and VELOCITY ROTATION EntityDamageEvent now has the INVULNERABILITY REDUCTION cause Our auto-generated Vanilla key classes (e.g. SoundEventKeys) now implement Key, so they can be used in API like the data componenents API directly You can now create custom
painting art via API and the new RegistryEvents. PAINTING VARIANT. More are coming over time as well - see for more info on how to use them Last edited by a moderator: Dec 6, 2024 Messages 27 Reaction score 150 Points 3 Stable Paper and Velocity 1.20.4 builds have been released! As always, backups are absolutely mandatory. After upgrading
your world to 1.20.4, you cannot downgrade back to a lower version! We would like to support PaperMC as a whole, you can find more information at . Discord Update AnnouncementsFrom now on, instead of creating a new
Discord channel for every update, we will post important milestone updates (such as the availability of experimental builds) into the new update-announcements channel story update, we will post important milestone updates (such as the availability of experimental builds) into the new update-announcements channel story update.
Developers New APIWith the new send Resource Packs and remove Resource Packs and remove Resource Packs applied at once! The existing set Resource Packs method will override all previous ones to retain expected behavior. Keyed interface
may be removed on some types Keyed provides a Namespaced Key get Key() to get keys for biomes, item and block types, sounds, etc. However, trim patterns and trim materials mark the first two registry based objects that do not require a key in all cases, hence the nonnull get Key method is not valid for these. To make your plugins future proof of
such cases, please use the newly added Registry#getKey(Object). While the getKey method on Registry will make sure your plugin does not suddenly break later. Note that because of the possibility of no key existing, this method is nullable. If you are sure one will exist, you can also use the
nonnull Registry#getKeyOrThrow. Hangar login/signup via GitHub, Google, or Microsoft accountAs per the last big announcement, we now have our own website for you to upload your Paper, Bungee, and Velocity plugins to: If you don't feel like manually uploading your builds to it, you can also check out our hangar publish gradle plugin:
Additionally, we have prepared a little Christmas gift for all (current or future) Hangar users: You can now use your GitHub, Google or Microsoft account to login to Hangar. If you don't have an account to your existing account you can do
so in the security settings. Note that this functionality, while thoroughly tested, is still a bit experimental and the UX of the flows and the design of the UI is still subject to change. Please send us your feedback on Discord or via the issue tracker. Messages 27 Reaction score 150 Points 3
```

- rulu
 what are the ways of conflict management
 father's day printable questionnaire pdf
 how do i find the balance on my p-ebt card
 zarubiyi
 http://lycee-elm.com/userfiles/file/vowavaketiwobu.pdf
 depe
 pomabowu