# How to: Improve All The Mods 9 Performance & stability

## Objective:

Provide installation guides and info about: Prism launcher, Java 17 and game client optimization.

Increase ATM9 stability & performance.

Free the people from the Curseforge prison.

*** Results not guaranteed**

## Methodology:

### 1. Abandon curseforge

### Why?

Curseforge is very bloated and lacks a lot of basic functionality.

It only carries curseforge packs, meaning you need to use multiple launchers if you wish to play eg. Tekkit.

It's awkward to use and tweaking modpacks is just painful with it.

## Prism?

Prism provides robust modpack management.

Allows you to play essentially all modpacks with one launcher.

Solid logging & debug.

Multi account/instance support.

Does not eat your ram.

It just works™

**Prism images & download link**

Supports multiple modpack sources (begone the suffering of installing 9023819 launchers)

image.png

Ability to assign instances their own settings (Maybe you want to use Java version X for one modpack and Java version Y for another, Ram allocations etc)

image.png

image.png

image.png

And of course you can also assign global options, in the case you'd want your modpacks to always launch with eg. 1920*1080 windowed mode, use X amount of ram & much more by default.

image.png

Readable logs directly in launcher

image.png

Mod, saves, resource packs, shader & more manageable directly in launcher

image.png

Visit their site at https://prismlauncher.org/ for further information & install options.

*Prism is a fork of MultiMC (that actually gets updates!)*

## 2. Install ATM9 with Prism

Time to smile.

### Adding ATM9 to Prism

1. In the setup of Prism, you can set these options how you see fit, this will become the **default global settings**, but for sake of simplicity, set the ram to anything you want, as we will set the ram allocation on the **instance setting** of ATM9 instead, same for the Java options - Just untick it and ignore anything it tells you because we will set it per instance anyway, this is covered in

the Java section later on.

image.png

2. Once installed & ready click the 'Add Instance' button, find curseforge and pick ATM9.

image.png

image.png

> When you first install ATM9 with the prism launcher, it's possible you'll be greeted with a window saying 'Mods not found' This is easy to resolve, just click the curseforge link in the window to download the mod and then drag it onto the window or click the 'add download folder' button and select the folder that the mod was downloaded to, then the red □ will turn into a □ and you can proceed.

image.png

Change any other settings if you want and/or move on step 3.

# 3. Java

Baseline for Minecraft 1.20.1 is Java version 17, so we will stick to it to ensure stability.

This will cover:

A. Oracle GraalVM Enterprise for Java 17

B. Adoptium OpenJDK 17

> I personally use GraalVM EE and have never had any bad experiences with it, I do recommend giving it a try - however if GraalVM does not work, just use Adoptium OpenJDK, It's essentially just as fast but less **experimental** than GraalVM.

**PRE**

I assign 12GB Ram to my ATM9 Instance in prism, for me this has been a solid number for over 2000> hours of gameplay, if it's possible, assign the same.

However if you want to use less or more I don't recommend going below 9GB or above 14GB, I have ran the game without crashes on 8,5GB but it was not a great experience - Just as not enough ram can cause issues, so will assigning too much ram, I recommend sticking to 12GB, you can set it to 14GB if you install the Distant Horizons mod.

> If your PC only have 16GB ram, this modpack could be tricky to run on Windows.

**Option A. GraalVM EE 21 for Java 17**

## Install GraalVM EE

GraalVM Docs can be found here

1. Download GraalVM here & follow the steps in the image (click the icon the emoji is pointing to) **note** that it requires you to create an account (It's free however.)

image.png

2. Once downloaded you'll end up with this zip file, use the archive tool of your choice to unpack it.

If you don't have an archive tool, you can get 7zip here

image.png

3. Once extracted, put the folder anywhere it'll be safe, I store them like this in the documents folder.

image.png

4. To use GraalVM in prism all you have to do is rightclick a modpack (or set it in the global options) and point it to the javaw.exe in the bin/ directory of the folder we extracted from the zip file.

Below is an example path to illustrate the bin/javaw.exe:

C:/Users/EXAMPLE/Documents/Java/graalvm-ee-java17-windows-amd64-x.x.x/**bin/javaw.exe**

Also tick the 'Skip Java compatibility checks' box.

image.png

image.png

image.png

5. To add launch arguments just tick the 'Java arguments' box and copy paste **one** of the options below.

image.png

**Note:** -XX:ReservedCodeCacheSize=xG will make the game use additional memory from the already assigned memory, use the 2GB option if you got ram to spare, else the 1GB.

**2GB copy & paste**

```
-XX:+UnlockExperimentalVMOptions -XX:+UnlockDiagnosticVMOptions -
XX:+AlwaysActAsServerClassMachine -XX:+AlwaysPreTouch -XX:+DisableExplicitGC -
XX:+UseNUMA -XX:AllocatePrefetchStyle=3 -XX:ReservedCodeCacheSize=2G -
XX:NonNMethodCodeHeapSize=64M -XX:ProfiledCodeHeapSize=1408M -
XX:NonProfiledCodeHeapSize=576M -XX:-DontCompileHugeMethods -
XX:+PerfDisableSharedMem -XX:+UseFastUnorderedTimeStamps -
XX:+UseCriticalJavaThreadPriority -XX:+EagerJVMCI -XX:+UseJVMCICompiler -
Dgraal.TuneInlinerExploration=1 -Dgraal.CompilerConfiguration=enterprise -XX:+UseG1GC
-XX:MaxGCPauseMillis=40 -XX:G1HeapRegionSize=16M -XX:G1NewSizePercent=25 -
XX:G1ReservePercent=15 -XX:SurvivorRatio=8 -XX:G1MixedGCCountTarget=4 -
XX:G1HeapWastePercent=10 -XX:InitiatingHeapOccupancyPercent=20 -
XX:G1RSetUpdatingPauseTimePercent=1 -XX:MaxTenuringThreshold=1 -
XX:G1ConcMarkStepDurationMillis=5 -XX:G1ConcRSHotCardLimit=16 -
XX:G1ConcRefinementServiceIntervalMillis=50 -XX:GCTimeRatio=99 -XX:+EnableJVMCI
```

### 1GB copy & paste

```
-XX:+UnlockExperimentalVMOptions -XX:+UnlockDiagnosticVMOptions -
XX:+AlwaysActAsServerClassMachine -XX:+AlwaysPreTouch -XX:+DisableExplicitGC -
XX:+UseNUMA -XX:AllocatePrefetchStyle=3 -XX:ReservedCodeCacheSize=1G -
XX:NonNMethodCodeHeapSize=32M -XX:ProfiledCodeHeapSize=768M -
XX:NonProfiledCodeHeapSize=224M -XX:-DontCompileHugeMethods -
XX:+PerfDisableSharedMem -XX:+UseFastUnorderedTimeStamps -
XX:+UseCriticalJavaThreadPriority -XX:+EagerJVMCI -XX:+UseJVMCICompiler -
Dgraal.TuneInlinerExploration=1 -Dgraal.CompilerConfiguration=enterprise -XX:+UseG1GC
-XX:MaxGCPauseMillis=40 -XX:G1HeapRegionSize=16M -XX:G1NewSizePercent=25 -
XX:G1ReservePercent=15 -XX:SurvivorRatio=8 -XX:G1MixedGCCountTarget=4 -
XX:G1HeapWastePercent=10 -XX:InitiatingHeapOccupancyPercent=20 -
XX:G1RSetUpdatingPauseTimePercent=1 -XX:MaxTenuringThreshold=1 -
XX:G1ConcMarkStepDurationMillis=5 -XX:G1ConcRSHotCardLimit=16 -
XX:G1ConcRefinementServiceIntervalMillis=50 -XX:GCTimeRatio=99 -XX:+EnableJVMCI
```

### Explanation of the launch arguments

| Launch Argument | Explanation |
|---|---|
| -XX:+UnlockExperimentalVMOptions | Enables experimental JVM options for advanced tuning. |
| -XX:+UnlockDiagnosticVMOptions | Enables diagnostic JVM options for debugging. |

| | |
|---|---|
| -XX:+AlwaysActAsServerClassMachine | Optimizes performance by forcing server-class machine behavior. |
| -XX:+AlwaysPreTouch | Pre-allocates all memory pages at startup to reduce runtime latency. |
| -XX:+DisableExplicitGC | Disables explicit calls to System.gc(). |
| -XX:+UseNUMA | Optimizes memory allocation for NUMA systems. |
| -XX:AllocatePrefetchStyle=3 | Adjusts prefetch style for modern processors. |
| -XX:ReservedCodeCacheSize=2G | Allocates 2 GB for JIT-compiled code storage. |
| -XX:NonNMethodCodeHeapSize=64M | Reserves 64 MB for auxiliary compiled code. |
| -XX:ProfiledCodeHeapSize=1408M | Allocates 1.375 GB for frequently executed code. |
| -XX:NonProfiledCodeHeapSize=576M | Sets 576 MB for less frequently executed code. |
| -XX:-DontCompileHugeMethods | Allows the JIT compiler to compile large methods. |
| -XX:+PerfDisableSharedMem | Disables shared memory for performance data to enhance security. |
| -XX:+UseFastUnorderedTimeStamps | Enables fast, hardware-assisted timestamp generation. |
| -XX:+UseCriticalJavaThreadPriority | Enables critical thread priorities for Java threads. |
| -XX:+EagerJVMCI | Eagerly initializes the JVMCI for Graal JIT operations. |
| -XX:+UseJVMCICompiler | Uses the Graal JIT compiler for advanced optimizations. |
| -Dgraal.TuneInlinerExploration=1 | Increases Graal's inlining exploration for better optimizations. |
| -Dgraal.CompilerConfiguration=enterprise | Enables enterprise-level optimizations in Graal. |
| -XX:+UseG1GC | Uses the Garbage-First (G1) Garbage Collector. |
| -XX:MaxGCPauseMillis=40 | Targets 40 ms for maximum garbage collection pauses. |
| -XX:G1HeapRegionSize=16M | Sets G1 heap region size to 16 MB for finer control. |
| -XX:G1NewSizePercent=25 | Sets 25% of the heap for the young generation. |
| -XX:G1ReservePercent=15 | Reserves 15% of the heap as free space for GC overhead. |
| -XX:SurvivorRatio=8 | Adjusts Eden-to-Survivor ratio to 8:1. |
| -XX:G1MixedGCCountTarget=4 | Targets 4 mixed garbage collection cycles. |

| | |
|---|---|
| -XX:G1HeapWastePercent=10 | Allows up to 10% heap waste before triggering GC. |
| -XX:InitiatingHeapOccupancyPercent=20 | Starts concurrent GC at 20% heap occupancy. |
| -XX:G1RSetUpdatingPauseTimePercent=1 | Allocates 1% of GC pause time to remembered set updates. |
| -XX:MaxTenuringThreshold=1 | Moves objects to the old generation after one young GC cycle. |
| -XX:G1ConcMarkStepDurationMillis=5 | Limits concurrent marking steps to 5 ms each. |
| -XX:G1ConcRSHotCardLimit=16 | Caps the number of hot cards processed during refinement. |
| -XX:G1ConcRefinementServiceIntervalMillis=50 | Sets concurrent refinement thread interval to 50 ms. |
| -XX:GCTimeRatio=99 | Targets 99% application time vs. 1% GC time. |
| -XX:+EnableJVMCI | Enables JVM Compiler Interface for Graal. |

## Done, Now you're ready to play the game and have fun, or optionally continue to the mods chapter.

### Option B. Adoptium OpenJDK for Java 17

## Install Adoptium OpenJDK 17

Download Adoptium OpenJDK here

Adoptium Docs can be found here

1. Download the installer by clicking the .msi button on the download site.

image.png

2. Run the installer and just click 'next' on everything.

image.png

image.png

3. To use Adoptium OpenJDK in prism all you have to do is rightclick a modpack (or set it in the global options) and click the 'Auto Detect' button.

image.png

image.png

4. Select it in the menu & tick the 'Skip Java compatibility checks' box

image.png

5. To add launch arguments just tick the 'Java arguments' box and copy paste one of the options below.

image.png

> **Note:** -XX:ReservedCodeCacheSize=xG will make the game use additional memory from the already assigned memory, use the 2GB option if you got ram to spare, else the 1GB.

**2GB copy & paste**

```
-XX:+UnlockExperimentalVMOptions -XX:+UnlockDiagnosticVMOptions -
XX:+AlwaysActAsServerClassMachine -XX:+AlwaysPreTouch -XX:+DisableExplicitGC -
XX:+UseNUMA -XX:AllocatePrefetchStyle=3 -XX:ReservedCodeCacheSize=2G -
XX:NonNMethodCodeHeapSize=64M -XX:ProfiledCodeHeapSize=1408M -
XX:NonProfiledCodeHeapSize=576M -XX:-DontCompileHugeMethods -
XX:+PerfDisableSharedMem -XX:+UseFastUnorderedTimeStamps -
XX:+UseCriticalJavaThreadPriority -XX:+UseG1GC -XX:MaxGCPauseMillis=40 -
XX:G1HeapRegionSize=16M -XX:G1NewSizePercent=25 -XX:G1ReservePercent=15 -
XX:SurvivorRatio=8 -XX:G1MixedGCCountTarget=4 -XX:G1HeapWastePercent=10 -
XX:InitiatingHeapOccupancyPercent=20 -XX:G1RSetUpdatingPauseTimePercent=1 -
XX:MaxTenuringThreshold=1 -XX:G1ConcMarkStepDurationMillis=5 -
```

```
XX:G1ConcRSHotCardLimit=16 -XX:G1ConcRefinementServiceIntervalMillis=50 -
XX:GCTimeRatio=99
```

### 1GB copy & paste

```
-XX:+UnlockExperimentalVMOptions -XX:+UnlockDiagnosticVMOptions -
XX:+AlwaysActAsServerClassMachine -XX:+AlwaysPreTouch -XX:+DisableExplicitGC -
XX:+UseNUMA -XX:AllocatePrefetchStyle=3 -XX:ReservedCodeCacheSize=1G -
XX:NonNMethodCodeHeapSize=32M -XX:ProfiledCodeHeapSize=768M -
XX:NonProfiledCodeHeapSize=224M -XX:-DontCompileHugeMethods -
XX:+PerfDisableSharedMem -XX:+UseFastUnorderedTimeStamps -
XX:+UseCriticalJavaThreadPriority -XX:+UseG1GC -XX:MaxGCPauseMillis=40 -
XX:G1HeapRegionSize=16M -XX:G1NewSizePercent=25 -XX:G1ReservePercent=15 -
XX:SurvivorRatio=8 -XX:G1MixedGCCountTarget=4 -XX:G1HeapWastePercent=10 -
XX:InitiatingHeapOccupancyPercent=20 -XX:G1RSetUpdatingPauseTimePercent=1 -
XX:MaxTenuringThreshold=1 -XX:G1ConcMarkStepDurationMillis=5 -
XX:G1ConcRSHotCardLimit=16 -XX:G1ConcRefinementServiceIntervalMillis=50 -
XX:GCTimeRatio=99
```

### Explanation of the launch arguments

| Launch Argument | Explanation |
|---|---|
| -XX:+UnlockExperimentalVMOptions | Enables experimental JVM options for advanced tuning. |
| -XX:+UnlockDiagnosticVMOptions | Enables diagnostic JVM options for debugging. |
| -XX:+AlwaysActAsServerClassMachine | Optimizes performance by forcing server-class machine behavior. |
| -XX:+AlwaysPreTouch | Pre-allocates all memory pages at startup to reduce runtime latency. |
| -XX:+DisableExplicitGC | Disables explicit calls to System.gc(). |
| -XX:+UseNUMA | Optimizes memory allocation for NUMA systems. |
| -XX:AllocatePrefetchStyle=3 | Adjusts prefetch style for modern processors. |
| -XX:ReservedCodeCacheSize=2G | Allocates 2 GB for JIT-compiled code storage. |
| -XX:NonNMethodCodeHeapSize=64M | Reserves 64 MB for auxiliary compiled code. |

| | |
|---|---|
| -XX:ProfiledCodeHeapSize=1408M | Allocates 1.375 GB for frequently executed code. |
| -XX:NonProfiledCodeHeapSize=576M | Sets 576 MB for less frequently executed code. |
| -XX:-DontCompileHugeMethods | Allows the JIT compiler to compile large methods. |
| -XX:+PerfDisableSharedMem | Disables shared memory for performance data to enhance security. |
| -XX:+UseFastUnorderedTimeStamps | Enables fast, hardware-assisted timestamp generation. |
| -XX:+UseCriticalJavaThreadPriority | Enables critical thread priorities for Java threads. |
| -XX:+UseG1GC | Uses the Garbage-First (G1) Garbage Collector. |
| -XX:MaxGCPauseMillis=40 | Targets 40 ms for maximum garbage collection pauses. |
| -XX:G1HeapRegionSize=16M | Sets G1 heap region size to 16 MB for finer control. |
| -XX:G1NewSizePercent=25 | Sets 25% of the heap for the young generation. |
| -XX:G1ReservePercent=15 | Reserves 15% of the heap as free space for GC overhead. |
| -XX:SurvivorRatio=8 | Adjusts Eden-to-Survivor ratio to 8:1. |
| -XX:G1MixedGCCountTarget=4 | Targets 4 mixed garbage collection cycles. |
| -XX:G1HeapWastePercent=10 | Allows up to 10% heap waste before triggering GC. |
| -XX:InitiatingHeapOccupancyPercent=20 | Starts concurrent GC at 20% heap occupancy. |
| -XX:G1RSetUpdatingPauseTimePercent=1 | Allocates 1% of GC pause time to remembered set updates. |
| -XX:MaxTenuringThreshold=1 | Moves objects to the old generation after one young GC cycle. |
| -XX:G1ConcMarkStepDurationMillis=5 | Limits concurrent marking steps to 5 ms each. |
| -XX:G1ConcRSHotCardLimit=16 | Caps the number of hot cards processed during refinement. |
| -XX:G1ConcRefinementServiceIntervalMillis=50 | Sets concurrent refinement thread interval to 50 ms. |
| -XX:GCTimeRatio=99 | Targets 99% application time vs. 1% GC time. |

Done, Now you're ready to play the game and have fun, or optionally continue to the mods chapter.

# 4. Mods (Optional)

This are my personal takes + some mod recommendations/info

## Mods I use & recommend

### EMI

Really recommend using this mod instead of JEI, it provides a nice configurable UI, Recipe trees, faster indexing & baking and more.

Installation steps: just drop the .jar file in the ATM9 mods folder, nothing else needs to be done.

*I also recommend enabling 'Batched Renderer' in the EMI options.

Curseforge link here

### Embeddium++

Great addon to the embeddium mod, gives you additional graphics options and allows you to display some diagnostics like FPS, ram useage etc

Installation steps: just drop the .jar file in the ATM9 mods folder, nothing else needs to be done.

Curseforge link here

### MacroKeybinds

I enjoy binding commands to keys, if you also enjoy it then go ahead

Installation steps: just drop the .jar file in the ATM9 mods folder, nothing else needs to be done.

Curseforge link here

## Mods that can be removed

### InvTweaks

Pointless inventory addon mod that adds a button to the side of various container GUI's (it's meant to be used for misc sorting actions but it has a feature of instantly crashing your game if accidentally pressed in some container GUI's that don't support it.) + The button is real ugly.

Can safely be deleted from the mods folder.

### Oculus

If you plan on using the Distant Horizons mod OR wont be using any shaders, you can delete it from the mods folder.

### Observable & Crashutilities

Two mods used for server administration & performance monitoring, cannot be used by non OP players so can safely remove them.

## Info

### Leftovers

- If you don't like how Modernfix makes the Gregtech textures look, you can remove the mod, beware that this will slightly increase ram useage and in some instances lead to game instabilities, it causes no issues otherwise and can stay.

- If you want to disable the RPS (the game activity discord thing that says "playing atm9") set discord to false in allthetweaks-common.toml in the config/ directory of the modpack instance folder.

image.png

- If you have issues with random disconnects / timeouts - try using the Play.SiriusMC.net IP instead of ATM9.SiriusMC.net, alternatively you can try using a VPN, just make sure to get a trustworthy and reputable provider.

> Notify an Admin or higher on the SiriusMC Discord if you wish to try playing with a VPN (VPN connections are by default blocked)

#### HOPEFULLY YOUR GAME RUNS WELL NOW AND YOU CAN SMILE MORE, FEEL FREE TO CONTACT ME (LOGICNFACTS) ON THE SIRIUSMC DISCORD IF YOU HAVE ANY QUESTIONS.

Revision #9
Created 6 December 2024 00:54:07 by LOGICNFACTS
Updated 7 December 2024 16:14:55 by LOGICNFACTS