Set up Garry’s Mod Workshop collections and gamemodes

Garry's Mod uses Source 1. Its Workshop collection startup option is different from CS2's map command, and a collection does not automatically select a gamemode. Keep collection loading, gamemode selection and client content downloads as three separate checks.

Prepare a small collection

Back up your startup settings, configuration and addon data first. Record the last working map and gamemode. Make changes during a maintenance window, because a full restart is usually the clearest way to confirm what mounted at startup.

Create a collection containing the intended addons and their required dependencies. Make it public or unlisted so the dedicated server can access it. Record the collection's numeric ID from its Workshop URL. Check that you copied a collection ID rather than an individual addon ID, and review each item's required content and supported gamemode.

Start with only the essentials. A large collection makes first startup slower and gives you many possible causes when an addon fails. Keep a dated list of the items you have tested, especially when several staff members can edit the collection.

Mount content and select the mode

1. Set the service's collection value if its startup template provides one. Otherwise ask support to add `+host_workshop_collection COLLECTION_ID` to the managed startup configuration.
2. Select the gamemode using its internal folder name, such as `+gamemode sandbox`. A Workshop item number is not a gamemode name.

3. Use the installed map name without `.bsp`, for example `+map gm_construct`, while preserving your assigned network settings.
4. Restart normally and read the Workshop mounting output, gamemode initialization and map load messages.

A manually installed gamemode belongs under `garrysmod/gamemodes/`, with its expected internal directory structure. Avoid accidentally creating two nested copies of the same gamemode folder. Workshop-managed content should remain managed through the collection rather than being duplicated as extracted local addons.

Check what players receive

Mounting an addon on the server and telling clients to download its assets are different tasks. Facepunch's `resource.AddWorkshop` marks an item for client download; it does not install that item on the server. A server-side Lua file can declare the required client content item IDs.

Do not indiscriminately mark every collection entry for client download. A server-only Lua addon may not need to be downloaded as Workshop content. Check the author's instructions for models, materials, sounds and gamemode resources. The active Workshop map has additional automatic handling, but other assets still need checking.

Verify the complete experience

Join with a test client that has not already subscribed to your full collection. Confirm the correct mode is active, the map loads, required assets appear and basic gameplay works. A staff member with every addon installed can miss a broken download configuration.

If it fails, keep the first error and item ID. Return to the saved baseline, then introduce dependencies and addons in small groups. Preserve data folders and external databases throughout testing. Do not wipe Workshop caches, saves or all addons to fix a single incompatible item; narrow down the failing content and keep a reversible record of each change.

Related guides

- Fix missing Garry’s Mod content
- Isolate Garry’s Mod Lua errors

Further reading

- Facepunch: Workshop for dedicated servers
- Facepunch: dedicated server setup
- Facepunch: resource.AddWorkshop

Looking for another game? Browse game server setup and troubleshooting guides.

Revision #2
Created 2026-11-09 09:21:20 UTC by CloudExa Hosting Admin
Updated 2026-11-09 09:21:21 UTC by CloudExa Hosting Admin