Guide contents

Build a local addon first

Add content or client behavior deliberately

Package for Workshop

Sources / References

Documentation: In progress **Quick start: Complete**

Offline copy from ModDocs. Source links remain clickable.

Garry's Mod addon creation

Garry's Mod loads **addons** from folders under `garrysmod/addons`. An addon can contain Lua, maps, models, materials, sounds or a combination. Start with one small Lua feature in a local addon, then package that folder for the Workshop only after it works in a clean game. The Facepunch addon guide and Lua folder reference are the current source documents.

> **Quick start:** create a lowercase addon folder containing one server-side Lua file. Register a unique console command, start a local game, run the command and confirm its result. Check that removing the addon removes the command.

Build a local addon first

Use Steam's **Browse local files** to find the installed game. Make the following layout under `garrysmod`:

```text
addons/moddocs_hello/
  lua/autorun/server/moddocs_hello.lua
```

The addon structure page says addon folders mirror the game's own folders. `lua/autorun/server` loads on the server; the sibling `client` folder loads only client-side, while `lua/autorun` files are shared. Keep this first command server-side so its

behavior and security checks have one location. The addon folder name should use lowercase letters, numbers, underscores or dashes.

In `moddocs_hello.lua`, register a unique command:

```lua
concommand.Add("moddocs_hello", function(ply)
    if IsValid(ply) then
        ply:ChatPrint("Hello from the addon.")
    else
        print("Hello from the addon console.")
    end
end)
```

The console-command API documents the callback's player argument and warns that a server command can still be invoked by clients. This example only sends a message; an admin command would require a permission check. Start a local game, open the console, run `moddocs_hello` and look for the chat message. A dedicated-server console invocation should print there instead. If the command is unknown, check the exact folder nesting, file extension and server Lua errors.

Add content or client behavior deliberately

For UI, HUD or effects, place code in the correct client/shared realm and use the realm documentation before passing data between server and client. A file running server-side does not automatically make its functions available to the client. Add models under `models`, materials under `materials`, and sounds under **`sound`**, not `sounds`; the addon guide lists the mounted folders. A map has its own BSP and optional navigation resources. Add only the folders your addon actually needs.

Test with other addons disabled when diagnosing a name collision or override. After each feature, restart the game and verify both that your addon loads and that the player-visible effect works. For multiplayer, connect a second client; local single-player success does not prove that server assets are delivered correctly. The content-serving guide covers Workshop downloads for servers.

Package for Workshop

The Workshop creation guide documents the current file whitelist, `addon.json`, `gmad` and `gmpublish` workflow. Test the addon folder in `garrysmod/addons` first, then package the same folder as a `.gma`. Make an appropriate baseline JPEG icon if publishing. Check the whitelist before including build scripts or source files: executable and certain document extensions are restricted. Upload only content you have rights to share. Once published, test the Workshop copy with your loose local folder removed so it cannot mask a missing asset.

The command and folder form a complete small local addon, and the Workshop section covers packaging its tested copy. The walkthrough is source-reviewed against the linked Facepunch documentation; it has not been run in a local Garry's Mod installation here. Weapon, entity, gamemode, map and advanced networking projects require their own APIs, so use the linked references for those larger features.

Sources / References

External project and documentation links cited in this guide. Check the upstream release and version before following version-sensitive steps.

- Facepunch addon guide
- Lua folder reference
- console-command API
- realm documentation
- content-serving guide
- Workshop creation guide