Guide contents

Install the loader and confirm it works

Make the mod discoverable

Add one game object

Test, package and extend

Sources / References

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

Offline copy from ModDocs. Source links remain clickable.

Balatro modding with Steamodded

Balatro's community mod route uses **Lovely** to patch the game and **Steamodded (SMODS)** to load Lua mods and register game objects. The Steamodded first-mod guide is the starting point; use the version of that documentation matching the loader you install. Keep an untouched copy of saves before testing a new mod.

> **Quick start:** install Lovely and Steamodded using their current instructions, create a mod folder with JSON metadata and `main.lua`, confirm it appears in the Mods menu, then register one Joker and test it in a run.

Install the loader and confirm it works

Follow the Steamodded installation instructions for your operating system. On Windows, the documented mods location is `%AppData%/Balatro/Mods`. The guide supports cloning Steamodded into that folder or following its manual installer; both require Lovely. Launch Balatro once with only the loader installed. Confirm that its Mods interface appears before diagnosing your own Lua.

Do not put a mod next to the game's executable merely because that seems familiar from other games. Steamodded searches the Balatro mods directory. Keep a separate folder for each mod so dependencies and updates can be isolated.

Make the mod discoverable

The current metadata specification prefers a separate JSON file. A minimal `Mods/moddocs_test/moddocs_test.json` can contain:

```json
{
  "id": "moddocs_test",
  "name": "ModDocs Test",
  "author": ["Your Name"],
  "description": "A small test mod",
  "prefix": "mdt",
  "main_file": "main.lua",
  "version": "1.0.0"
}
```

The `id` and `prefix` must be unique across installed mods; `main_file` must point to a real file in the folder. This is valid JSON: avoid copying comments or trailing commas from explanatory examples. Put a `main.lua` file beside the JSON. Do not also add the old `--- STEAMODDED HEADER` to that file: the metadata page warns that mixing the two discovery methods can load a mod twice.

Start with a simple Lua log statement in the entry file and confirm the mod appears in-game. If it does not, check the loader version, path, JSON syntax, file name and log before adding gameplay behavior.

Add one game object

Steamodded's game-object API includes Jokers, consumables, decks, blinds, vouchers and more. A Joker is registered with `SMODS.Joker` and needs a unique key plus localized text according to the Joker reference. Its metadata prefix is added to the resulting object key, so refer to the full key when connecting other objects.

Here is a complete **gameplay mod** using the modern `SMODS.Joker { ... }` API. First close Balatro and back up `%AppData%/Balatro` to another folder. Keep that backup with the date and your game, Lovely, and Steamodded versions. The example was reviewed against Steamodded documentation on **2026-10-05**; ModDocs has **not independently run it in the game** and therefore does not claim a verified version pair. Guide revision: **0.2**. It is meant for the PC version of Balatro and adds a Common

Joker that scores +4 Mult. It does not modify the executable or provide any copyrighted game assets.

Create this exact tree. Windows Explorer may hide extensions, so turn on **View → File name extensions** and ensure the files are not accidentally named `.json.txt` or `.lua.txt`.

```text
%AppData%/Balatro/Mods/
├── smods/                      Steamodded loader files
└── moddocs_workshop/
    ├── moddocs_workshop.json
    └── main.lua
```

Use this valid JSON in `moddocs_workshop.json`:

```json
{
  "id": "moddocs_workshop",
  "name": "ModDocs Workshop Joker",
  "author": ["Your Name"],
  "description": "A learning example that adds a +4 Mult Joker.",
  "prefix": "mdw",
  "main_file": "main.lua",
  "version": "0.1.0"
}
```

The `id` identifies the mod; the `prefix` is added to the keys of its registered objects; and `main_file` tells Steamodded which Lua file to execute. Keep the ID and prefix unique if you adapt the example. Do not copy comments or trailing commas into JSON. Do not add an old `--- STEAMODDED HEADER` to `main.lua` when using JSON metadata: the metadata specification warns that mixing the two discovery methods can load a mod twice.

Use this code in `main.lua`:

```lua
sendInfoMessage("Workshop Joker entry file loaded", "ModDocsWorkshop")

SMODS.Joker {
    key = "workshop_joker",
    loc_txt = {
        name = "Workshop Joker",
        text = { "{C:mult}+4{} Mult" }
    },
```

```
    rarity = 1,
    cost = 3,
    unlocked = true,
    discovered = true,
    blueprint_compat = true,
    calculate = function(self, card, context)
        if context.joker_main then
            return { mult = 4 }
        end
    end
}
```

`SMODS.Joker { ... }` registers the Joker at load time. `loc_txt` supplies its visible name and description without a separate translation file. `rarity = 1` means Common. `calculate` receives several contexts; `context.joker_main` restricts this effect to normal Joker scoring, and `{ mult = 4 }` asks Steamodded to add four Mult. This is based on the current Joker API and calculate reference. The example uses Steamodded's default Joker artwork so that no image file is required for its scoring behavior.

Check each stage in game

1. Fully restart Balatro. Open **Mods** and confirm **ModDocs Workshop Joker** appears. If it does not, fix discovery before changing the Lua code.
2. Look for `Workshop Joker entry file loaded` in the Lovely console or its logs. The logging reference locates log files under `%AppData%/Balatro/Mods/lovely/log`. This line confirms that the entry file ran, but does not alone prove the Joker registered.
3. Open the Joker collection and find **Workshop Joker**. If it is missing, read the first Lua error in the log.
4. Begin a disposable fresh run. The Common Joker can appear in ordinary shop selection. Obtain it, play a hand, and check for its **+4 Mult** scoring effect. Other cards can alter the final total, so inspect the individual Joker scoring step.
5. Save and reload that test run with the mod still installed. Confirm that the Joker remains and still scores. Record the game and loader versions you actually tested.

Optional original sprite

After the two-file mod works, draw **your own** 71 × 95 pixel PNG and save it as `assets/1x/workshop_joker.png`. For older Steamodded releases, also provide a doubled 142 × 190 image at `assets/2x/workshop_joker.png`. The atlas reference notes that releases since 26.829.0 can generate a missing resolution, but older releases may require both. Do not redistribute a sprite copied from Balatro without permission.

Add this block **above** the Joker registration in `main.lua`:

```lua
SMODS.Atlas {
    key = "workshop_art",
    path = "workshop_joker.png",
    px = 71,
    py = 95
}
```

Add `atlas = "workshop_art",` and `pos = { x = 0, y = 0 },` inside the Joker table after `key`. The position selects the top-left atlas frame. Restart and verify the image in the collection and shop, then retest scoring. The optional final tree is:

```text
moddocs_workshop/
├── moddocs_workshop.json
├── main.lua
└── assets/
    ├── 1x/workshop_joker.png
    └── 2x/workshop_joker.png
```

Test, package and extend

Test both a clean profile with only Lovely, Steamodded and your mod, and a second profile with the dependencies you claim. Record the Balatro, Lovely and Steamodded versions. Include your original `main.lua`, JSON metadata, localization and assets. If you add sprites, consult the current atlas documentation for dimensions and 1x/2x requirements. For distribution, the Steamodded packaging guide is a separate step from local testing.

Diagnose by symptom

Symptom	Diagnostic action
Lovely window absent	Check `winmm.dll` beside the Balatro executable and the exact Windows Lovely release. An old `version.dll` can conflict during an upgrade.
Lovely runs; Mods button absent	Inspect `%AppData%/Balatro/Mods/smods` for an extra nested `smods` folder. Follow the upstream Windows installation guide.
This mod absent from Mods	Verify both real file extensions, valid JSON, matching `main_file`, and a unique `id`. Read the first relevant log error.
Mod present; Joker absent	Confirm the entry-file log, then check for an exception in `SMODS.Joker`, `key`, or `loc_txt`. Do not mix JSON discovery with a legacy header.
Joker present; no +4 Mult	Obtain it in an active run; appearing in the collection does not score. Check `context.joker_main` and the `{ mult = 4 }` return, then test with other mods disabled.
Sprite blank after optional art step	Check exact PNG filename and case, `assets/1x` and `assets/2x`, dimensions, atlas key, and `{ x = 0, y = 0 }`. Temporarily remove the art fields to isolate rendering.
Old save fails after removing mod	Restore the exact mod used by that run or your pre-mod save backup. Use fresh disposable runs for compatibility tests.

To uninstall this example, close Balatro and move **only** `moddocs_workshop` out of `Mods`; leave Steamodded and other people's mods alone. Preserve your JSON, Lua, original art, this PDF/HTML guide, and the version IDs of the dependencies used for a working build. ModDocs does not mirror the game or loader binaries. For historical projects that use `SMODS.Joker:new(...):register()`, consult the Steamodded migration guide; that older call is not interchangeable with this modern example.

Sources / References

- Steamodded Windows installation and troubleshooting

- Steamodded first-mod guide
- Steamodded metadata specification
- SMODS.Joker API
- Calculate functions and scoring contexts
- Atlas API
- Steamodded logging
- Steamodded source and releases and Lovely source and releases