Page contents

- Preflight the project
- Publish tab and cooking
- Verify the installed package
- Release manifest and rollback

Offline copy of Cooking, installation and release checks .

Cooking, installation and release checks

CD PROJEKT RED's Publishing mods documents the REDkit packaging flow. This chapter separates building a local package from installing or publishing it. It is source-reviewed; no independent game test is recorded.

Preflight the project

Save the project and test the intended change from REDkit using the correct `.redgame` definition. Record the game build, REDkit build, project version, selected definition, test world and exact result. Resolve editor errors before cooking. For a quest or world edit, verify the relevant layers and scene transitions, not only the presence of a resource in the Asset Browser. Retain a backup of the prior project and a pre-test save.

Publish tab and cooking

Open the **Publish** tab and choose **Save and publish mod project**. The official wizard requests basic project details such as name, version, description and thumbnail. It then cooks the project: REDkit analyzes the required files and creates bundles, caches and metadata in a `packed` output area. Inspect the log for the first failure and the packed output for the files you intended to ship. A successful workspace save is distinct from a

successful cook. If a resource is absent, trace its project reference and virtual path before adding unrelated files to the package.

The wizard distinguishes exporting a ZIP, installing the project locally and publishing through Steam Workshop. Choose an export for a shareable file; use local installation for an installed-game check. Workshop submission is an external publication step and should follow completed testing and a review of the visible metadata. The official page also describes language and visibility choices. Document the exact choice for each release; a local ZIP is not evidence that a Workshop item exists.

Verify the installed package

Inspect the exported archive for the intended mod directory and cooked files. Install it into a disposable game copy or clean profile. First test with no unrelated mods. Confirm the change at a repeatable in-game location, record error messages, then test saving, loading and repeating the action where relevant. For a world or quest mod, verify that all required scenes and conditions survive a restart. Only then test combinations with other mods that touch the same resources. Record each compatible/incompatible combination with versions; do not generalize from one pair to all mods.

If the editor test passes but the installed game does not, compare project workspace, packed output and installed directory. A file present only in the workspace was not shipped. If a previous installed copy remains, remove that test copy before retrying and preserve the prior release separately. If a game update changes behavior, reproduce with the same mod archive and an exact game build before recooking; this isolates package drift from game drift.

Release manifest and rollback

Ship only authored or properly licensed content, the mod's required metadata, and clear installation instructions. Include a title, unique mod identifier, version, game/REDkit version, affected resources, dependencies, load/conflict notes, tested configurations, known limitations and rollback steps. A checksum for the final ZIP helps distinguish the published artifact from a later local rebuild. Keep the original project, source notes and prior release archive so an old version can be reconstructed or removed without touching player saves. The uncooked depot and original game installation are not release payloads.

Further source coverage remains open for managing mods, virtual depot and source control, DLC mods, and version-specific conflict rules. Treat those as research gaps until their actual pages are audited.