

FACTION CREATION TOOL

Workbench User Guide

A practical guide to preparing, configuring, building, and validating an Arma Reforger faction.

For mod creators using Arma Reforger Workbench

Guide edition 1.0 | 30 September 2026

QUICK OVERVIEW Create or load an FCT project, prepare your own prefab assets, edit/register them with the appropriate builders, generate catalogs and faction outputs, then validate the result.

Contents

- 1. What the tool does
- 2. Before you begin
- 3. Create or load your FCT project
- 4. Create the faction label
- 5. Prepare and edit assets
- 6. Generate catalogs and faction configuration
- 7. Build Arsenal and placeable support
- 8. Validate and run Final Build
- 9. Troubleshooting checklist

1. What the tool does

FCT coordinates faction configuration and project registration inside Workbench. Its builders edit supported properties on existing Reforger resources and generate project outputs.

- FCT can manage a faction project, edit supported prefab properties, register assets, build entity catalogs and a faction config, generate an Arsenal child prefab, build placeable registries, integrate EditorMode, and validate outputs.
- FCT does not create character models, vehicle models, weapon prefabs, or every asset from scratch. Prepare or copy the actual source prefabs into your target addon first.
- Feature scope is deliberate: for example, the Vehicle Builder focuses on faction affiliation and inventory MultiSlots; it is not a general vehicle physics or damage editor.

IMPORTANT A saved prefab and a project-registered/catalogued prefab are separate things. Follow the registration and catalog steps below; merely editing a prefab does not guarantee that Final Build will include it.

2. Before you begin

Have these ready before opening the builders:

- Arma Reforger Workbench with the Faction Creation Tool addon loaded.
- A separate, writable target addon where faction outputs and prefab edits can be saved. Do not use the FCT tool addon itself as the generated target.
- Source .et prefabs for your faction's characters, vehicles, groups, equipment, and weapons, plus any required base configs/resources, This includes a plan for what Dependencies you want. .
- A faction folder name, faction key, and display name. Use filesystem-safe folder names. Avoid path separators in the faction key.
- Open World Editor when using its prefab builders or any workflow that requires WorldEditorAPI.

TIP Keep your source asset addon and the target/generated faction addon distinct. Confirm the target path is writable before building.

3. Create or load your FCT project

Open Resource Manager > Plugins > Faction Creation Tool > FCT Project Manager.

1. Set Target Addon Root to the writable addon that will receive your faction outputs.
 2. Set Faction Folder Name, Faction Key, and Faction Display Name. Project Name may be left blank if you want it to default from the faction folder.
 3. Leave Auto Create Structure enabled unless you have already created the expected folders, then click CREATE PROJECT.
 4. Click SAVE PROJECT to write the project manifest. To continue later, use LOAD PROJECT and select that manifest.
- CREATE FOLDER STRUCTURE creates the expected faction asset folders under the target addon.
 - PRINT PATHS and PRINT SUMMARY are useful for checking the active project; VALIDATE PROJECT checks project metadata and registered data.

PROJECT LOCATION The saved project manifest is stored under the target addon's FCT/Projects/<FactionFolder>/ folder. Creating/loading a project makes it active for other FCT tools.

4. Create the faction label

In Resource Manager > Plugins > Faction Creation Tool > FCT Faction Editable Entity Label:

1. Confirm the correct project is active; click REFRESH PROJECT if needed.
 2. Click CREATE / SYNC FACTION LABEL. FCT creates or adopts the faction label entry named FACTION_<KEY> and stores its value in the project.
- Existing entries in the target addon label file are preserved. A conflicting numeric label value must be resolved before building.
 - Final Build also attempts this step, but creating the label early helps when authoring character prefabs.

5. Prepare and edit assets

Make or copy the source .et files into the target addon's expected category folders. Use categorized subfolders below each faction asset root; catalog scanning can skip prefabs placed directly at the faction root.

Characters

- Open one character prefab in Prefab Edit Mode, then open FCT Character Prefab Builder in World Editor.
- Click LOAD CURRENT VALUES. Enable the edit toggles for the sections you intend to change, then click SAVE CHANGES.
- Supported sections include display name, faction affiliation, rank, project-managed Editable Entity Labels, identity, weapon slots, worn equipment, and initial inventory.
- Use LOAD PROJECT DEFAULT LABELS to apply project-default label assignments. Equipment changes also rebuild inventory storage automatically.

CHARACTER REGISTRATION The Character Builder does not register characters in the project. Add the character ResourceName to the Catalog Builder's character inputs and click SYNC CATALOG INPUTS TO PROJECT.

Vehicles

- Select exactly one vehicle in World Editor and open FCT Vehicle Builder.
- Click LOAD CURRENT VALUES; use REFRESH / PRINT FACTIONS if faction choices need refreshing.
- Use ADD MULTISLOT / REMOVE LAST MULTISLOT as needed. Enable the MultiSlots edit option before saving changes.
- Click SAVE CHANGES. Registration and project autosave are enabled by default; review the tool fields before relying on defaults.

Groups

- Open the group prefab in Prefab Edit Mode and use FCT Group Builder.
- Click LOAD CURRENT VALUES. Use REFRESH / PRINT FACTIONS, ADD EMPTY UNIT SLOT, and REMOVE LAST UNIT SLOT as needed, then SAVE CHANGES.
- The builder edits SCR_AIGroup faction and unit prefab slots. Empty member slots are omitted on save. Registration and autosave default on.

Equipment

- Select one equipment prefab in World Editor and open FCT Equipment Builder. Click LOAD CURRENT VALUES, enable the sections to edit, then SAVE CHANGES.
- REGISTER EQUIPMENT is available separately. Registration and autosave default on.
- Editable properties depend on the prefab's supported components. Missing display-info, armor-storage, or modular-clothing components limit the corresponding edits.

Weapons

- Select one weapon prefab and open FCT Weapon Builder. Click LOAD CURRENT VALUES to discover supported name and existing attachment slots.
- Enable name and/or attachment-slot edits, then click SAVE CHANGES. Registration and autosave default on.

- The builder edits discovered existing slots; it is not a general weapon-stat editor and does not create attachment slots.

6. Generate catalogs and faction configuration

Entity catalogs

In Resource Manager > Plugins > Faction Creation Tool > FCT Catalog Builder:

1. Click REFRESH PROJECT to load the active project state.
2. Check the catalog input arrays. Add character ResourceNames that were edited but not registered by a builder.
3. Click SYNC CATALOG INPUTS TO PROJECT to persist the visible catalog inputs to the project registry.
4. Confirm the .et files are inside the expected category folders, then click BUILD ALL CATALOGS.
 - The builder creates Character, Vehicle, Group, and Inventory Item catalogs. Equipment and weapons are inventory items; Arsenal assets are excluded from that catalog.
 - The first subfolder below the faction asset root is used as the catalog category/list. Put assets into category subfolders, not directly in the faction root.

Faction config

In Resource Manager > Plugins > Faction Creation Tool > FCT Faction Config Builder:

1. Click LOAD PROJECT VALUES and review the base faction config, identity, callsign, territory, icon/flag, radio/friendly-faction settings, and four catalog references.
2. Set any required resource references, then click BUILD FACTION CONFIG.
 - The identity setting references an identity config; it does not generate an identity resource.
 - The builder requires an active project, faction label, and catalog references. Verify the generated config path and references if validation cannot find the output.

7. Build Arsenal and placeable support

Arsenal

- In World Editor, open FCT Arsenal Builder and click LOAD PROJECT VALUES, then BUILD ARSENAL.
- No entity selection is required, but World EditorAPI must be available. The builder creates or updates a faction-specific child of its configured source Arsenal and registers it as an Arsenal system asset by default.

Placeable registries and EditorMode

- In World Editor, open FCT Placeable Builder and click LOAD / REFRESH PROJECT.
- Click BUILD ALL PLACEABLES to generate Character, Vehicle, Group, and Systems/Arsenal registries and integrate them into the target addon's EditorModeEdit.et override.
- Existing EditorMode registry entries are preserved. FCT checks character/vehicle/group prefab editability; unsupported prefabs may be skipped with warnings.
- BUILD / UPDATE REGISTRIES ONLY and INTEGRATE EDITORMODE ONLY are available when you need to run just one part.

8. Validate and run Final Build

First run Resource Manager > Plugins > Faction Creation Tool > FCT Validator > VALIDATE FACTION PROJECT. Review all errors and warnings and resolve missing resources, references, or registration issues.

FINAL BUILD REQUIREMENTS Final Build requires an active project, a valid existing target addon root, and at least one registered Character, Vehicle, Group, and non-Arsenal equipment/weapon asset. It also requires World EditorAPI.

When prerequisites are met, open FCT Final Build in World Editor. No entity selection is required. Review the stage options, then click BUILD / VALIDATE / FINALIZE. The default order is:

1. Ensure the faction Editable Entity Label exists.
2. Build Entity Catalogs.
3. Build the faction config.
4. Build/update the faction Arsenal.
5. Build placeable registries and integrate EditorMode.
6. Run complete project validation and mark the project READY only if all stages succeed.
 - Final Build's counts are based on enabled project-registered assets. A prefab sitting in a folder without the required project registration may not satisfy its prerequisites.
 - FCT Project Manager > VALIDATE PROJECT is a separate project-data check; use the comprehensive FCT Validator for generated catalogs, Arsenal, placeables, and EditorMode outputs.

9. Troubleshooting checklist

Tool says no active project: Create or load the project in FCT Project Manager, then click REFRESH PROJECT in the relevant builder.

Asset is missing from a catalog: Confirm the .et file is under the correct target-addon root and a category subfolder; refresh/sync project inputs and rebuild catalogs.

Final Build says an asset category is empty: Register at least one enabled asset of that type. Characters need the Catalog Builder input + sync step; check the project registry for all categories.

Prefab values do not save: Confirm the correct prefab is open/selected, the required edit toggles are enabled, the target addon is writable, and the prefab has the component the builder edits.

World EditorAPI is unavailable: Run Arsenal, Placeable, and Final Build from World Editor, not Resource Manager.

Faction/config/catalog validation fails: Check the faction label exists, all catalog references point to saved resources, and generated resources are inside the target addon.

A group is rejected or warns: Open a prefab that inherits from SCR_AIGroup. Check its faction key and m_aUnitPrefabSlots values, then save and verify the prefab resource.

Changes are not reflected: Save the prefab and project, refresh the project state in the tool, then rebuild affected generated outputs.

Quick checklist

- [] Target addon exists, is writable, and is separate from the FCT tool addon.
- [] FCT project created/loaded and saved.

FACTION CREATION TOOL > USER GUIDE

- [] Faction label created/synced.
- [] Source prefabs prepared under categorized target-addon folders.
- [] Supported prefab fields edited and saved.
- [] Assets registered; character inputs synced to the project.
- [] All four catalogs built; faction config built.
- [] Arsenal built; placeable registries and EditorMode integrated.
- [] FCT Validator passes; Final Build completes successfully.