AuraSkills tasks

Task guides and troubleshooting for AuraSkills. Check the installed release before changing configuration or data.

- Separate an AuraSkills source reward from its XP multipliers
- Check AuraSkills player-placed block protection before allowing XP
- Find an AuraSkills source with the wrong material or source ID
- Check an AuraSkills menu that shows an outdated reward
- Test an AuraSkills money reward when Vault is unavailable
- Check recurring AuraSkills level rewards for off-by-one mistakes
- Distinguish an AuraSkills mana shortage from an ability cooldown
- Keep AuraSkills reward commands from targeting the wrong player

Separate an AuraSkills source reward from its XP multipliers

A changed skill reward can originate in its source definition or in a multiplier. Measure the base activity and player modifiers independently before adjusting progression across the server.

Check the installed content

AuraSkills connects skill XP sources to levels, rewards, abilities and menus. A change in one layer can leave another unchanged. Keep the existing skill identifiers and player data, and use the documentation matching your installed release when editing a source or reward.

Work through the change

1. Record the skill, source action and observed XP for an ordinary account. Save the relevant source file and note its permission groups, bonuses and other modifiers.
2. Find that source in the installed-release source reference. Treat the base source amount separately from the effective XP that appears after bonuses or sharing.
3. Repeat one controlled action on staging with a baseline account and the affected account. Keep tools, world and activity conditions identical so the comparison isolates player modifiers.
4. Change only the intended source or modifier and repeat after the release-supported configuration workflow. Include a second action from the same skill to check the scope of the change.

Verify the result

The selected action should produce the planned gain, and the comparison should explain why accounts differ. Other source rewards should retain their intended values.

If the result differs

If a source awards nothing, inspect eligibility and world restrictions before increasing its XP. If every source changes, look for a broader multiplier rather than changing each base amount independently.

Keep a recovery path

Restore the saved source and temporary permission changes if the result is unclear. Preserve skill data and avoid resetting levels to diagnose a future reward rate; stored progression is a separate concern.

References and related help

Check the maintainer reference: AuraSkills documentation, AuraSkills documentation.

Before changing live data, read how to keep a completed recovery copy. For startup errors, use the log collection guide.

Check AuraSkills player-placed block protection before allowing XP

Player-placed blocks may deliberately give no skill XP to prevent repeatable farming. Compare a natural block with a placed block before changing source eligibility.

Check the installed content

AuraSkills connects skill XP sources to levels, rewards, abilities and menus. A change in one layer can leave another unchanged. Keep the existing skill identifiers and player data, and use the documentation matching your installed release when editing a source or reward.

Work through the change

1. Record the affected skill, block and placement history. Prepare a small natural control area and a separate area containing blocks deliberately placed by the test account.
2. Read the source reference for the installed release and inspect its placement-check behaviour. The documented check_replace setting is enabled by default; verify the loaded configuration rather than assuming it changed.
3. Break the two sources under equivalent conditions with a normal account. Record the reward and compare any region or world restrictions that could affect both tests.
4. If your design needs a different policy, trial only the relevant source on staging. Repeat placement and breaking several times to see whether the proposed policy opens a simple farming loop.

Verify the result

The natural source should award XP where allowed, and the placed source should follow the intended protection rule. The outcome should be reproducible without modifying any player skill

levels.

If the result differs

If neither rewards XP, investigate skill access or source matching first. If another plugin replaces the blocks, compare that integration before disabling placement checks across the whole skill.

Keep a recovery path

Restore the protected source policy if the trial permits unintended repeatable gains. Keep the test blocks disposable and preserve player data; a source-policy adjustment should not erase legitimate progress.

References and related help

Check the maintainer reference: AuraSkills documentation.

Before changing live data, read how to keep a completed recovery copy. For startup errors, use the log collection guide.

Find an AuraSkills source with the wrong material or source ID

A source identifier and a Minecraft material name serve different purposes. Check both when a configured activity never awards XP or appears under an unexpected skill.

Check the installed content

AuraSkills connects skill XP sources to levels, rewards, abilities and menus. A change in one layer can leave another unchanged. Keep the existing skill identifiers and player data, and use the documentation matching your installed release when editing a source or reward.

Work through the change

1. Record the skill and the exact action that should award XP. Save its source file and compare the current definition with the source format for your installed AuraSkills release.
2. Identify the source entry ID separately from the material or entity it matches. Check the supported singular or plural field layout instead of adding both forms as a guess.
3. Create one minimal staging source using a documented supported target. Remove optional conditions from that test entry and perform the action with an ordinary account.
4. Add the intended target list and optional checks back one at a time. Read the first startup or configuration error after each change and repeat the same activity.

Verify the result

The intended activity should map to one loaded source and award the planned skill XP. Unrelated materials or entities should not receive rewards because a broad matcher was introduced.

If the result differs

If the source never loads, inspect YAML nesting and supported field names. If it loads but matches nothing, compare the real action and material identity before altering the skill or reward curve.

Keep a recovery path

Restore the previous source file and remove the disposable entry if matching remains uncertain. Keep player progression intact and retain the minimal working comparison so future edits do not reintroduce the same format error.

References and related help

Check the maintainer reference: AuraSkills documentation.

Before changing live data, read how to keep a completed recovery copy. For startup errors, use the log collection guide.

Check an AuraSkills menu that shows an outdated reward

A menu can describe an older reward while the actual level-up uses a newer definition. Compare presentation and reward execution before granting compensation or resetting a player level.

Check the installed content

AuraSkills connects skill XP sources to levels, rewards, abilities and menus. A change in one layer can leave another unchanged. Keep the existing skill identifiers and player data, and use the documentation matching your installed release when editing a source or reward.

Work through the change

1. Record the skill, level and exact menu text the player sees. Save the relevant menu and reward definitions together and note any translation or placeholder integration involved.
2. Read the menu and reward references for the installed release. Identify whether the displayed line is generated from reward data or is manually written descriptive text.
3. On staging, update only the incorrect presentation or reward reference. Compare the menu with the configured reward, then trigger a disposable test level-up through the supported test workflow.
4. Inspect what the test account actually receives and reopen the menu after relogging. Check neighbouring levels so a copied description does not misrepresent an entire reward interval.

Verify the result

The menu should describe the reward that the account actually receives at that level. Other skill pages and adjacent level descriptions should remain accurate and readable.

If the result differs

If the menu updates but the reward does not, inspect reward eligibility and dependencies. If the reward works but text remains old, trace cached menu or localisation content rather than editing player progress.

Keep a recovery path

Restore the saved menu and reward files if the trial introduces a mismatch. Keep real player levels unchanged and record any disposable test reward so it cannot enter the live economy accidentally.

References and related help

Check the maintainer reference: AuraSkills documentation, AuraSkills documentation.

Before changing live data, read how to keep a completed recovery copy. For startup errors, use the log collection guide.

Test an AuraSkills money reward when Vault is unavailable

A money reward depends on more than the skill level event. Check the economy bridge and provider before changing a reward amount that the player never receives.

Check the installed content

AuraSkills connects skill XP sources to levels, rewards, abilities and menus. A change in one layer can leave another unchanged. Keep the existing skill identifiers and player data, and use the documentation matching your installed release when editing a source or reward.

Work through the change

1. Record the reward level and configured amount, then save its reward definition. Confirm the installed-release documentation for the money reward and its Vault dependency.
2. On staging, inspect startup logs for AuraSkills, Vault and the actual economy provider. Vault is a bridge; its presence alone does not prove that an economy service registered successfully.
3. Trigger the reward with a disposable ordinary account and record its balance before and after. Compare another supported non-money reward at the same test level to locate the failing stage.
4. Repair the missing dependency or provider configuration on staging and restart through the normal server workflow. Repeat the same reward once without manually adding money during the comparison.

Verify the result

The account should receive the intended balance change through the registered economy provider, and the reward should not be issued twice by a separate compensation command.

If the result differs

If all rewards fail, inspect the level reward configuration first. If only money fails, review dependency registration and economy errors rather than increasing the amount or changing unrelated skill sources.

Keep a recovery path

Restore the previous reward definition if the dependency repair is not ready. Record the affected account and event for a separate reviewed compensation decision; configuration rollback cannot undo currency already paid.

References and related help

Check the maintainer reference: AuraSkills documentation.

Before changing live data, read how to keep a completed recovery copy. For startup errors, use the log collection guide.

Check recurring AuraSkills level rewards for off-by-one mistakes

An interval reward can appear at the wrong levels when its range or recurrence is misunderstood. List the intended levels before changing rewards already used by players.

Check the installed content

AuraSkills connects skill XP sources to levels, rewards, abilities and menus. A change in one layer can leave another unchanged. Keep the existing skill identifiers and player data, and use the documentation matching your installed release when editing a source or reward.

Work through the change

1. Write down the first few levels that should grant the reward and the levels immediately between them that should not. Save the skill reward file and note any overlapping reward definitions.
2. Compare the installed-release interval or level mapping format with the planned sequence. Keep a one-time level reward distinct from a repeating reward rule.
3. Build the smallest staging sequence that covers an eligible level, an ineligible level and the next recurrence. Use low-value observable rewards with a disposable player state.
4. Trigger each test transition through supported administration or normal progression and record the actual rewards. Include any overlapping rule so two grants are not mistaken for one interval firing twice.

Verify the result

The observed sequence should match the written level list, including its start and next recurrence. Adjacent levels should not receive an accidental grant from overlapping definitions.

If the result differs

If every level grants the reward, inspect its enclosing range or default rule. If only one transition differs, compare previous player state and whether the reward was already processed before changing the interval.

Keep a recovery path

Restore the previous reward mapping and disposable player checkpoint if the test sequence is wrong. Do not lower and raise real player levels repeatedly to diagnose reward timing, since that can create duplicate rewards.

References and related help

Check the maintainer reference: AuraSkills documentation.

Before changing live data, read how to keep a completed recovery copy. For startup errors, use the log collection guide.

Distinguish an AuraSkills mana shortage from an ability cooldown

An ability that does not activate may be waiting for mana or a cooldown rather than being broken. Test its prerequisites separately before lowering costs or removing limits.

Check the installed content

AuraSkills connects skill XP sources to levels, rewards, abilities and menus. A change in one layer can leave another unchanged. Keep the existing skill identifiers and player data, and use the documentation matching your installed release when editing a source or reward.

Work through the change

1. Record the skill, ability, current mana and the action used to activate it. Consult the installed-release mana-ability reference for the required tool, activation method and prerequisites.
2. Use an ordinary staging account that legitimately meets the skill requirement. Test once with enough mana and an expired cooldown, recording the feedback and effect.
3. Repeat immediately to observe cooldown behaviour, then test the insufficient-mana case separately. Keep other abilities and bonus equipment out of the comparison.
4. If adjusting the balance, change only the documented cost or cooldown setting and retest the same cases. Include a normal relog to confirm the display agrees with actual eligibility.

Verify the result

The ready account should activate the intended ability, while cooldown and low-mana cases should fail for their respective reasons. The displayed state should match the action the server permits.

If the result differs

If the fully ready case still fails, inspect tool eligibility, permissions and world restrictions. If activation works but the display lags, troubleshoot the display integration before changing the resource cost.

Keep a recovery path

Restore the prior ability settings if the trial changes balance unexpectedly. Keep existing player levels and mana-related data backed up, and remove temporary test grants instead of leaving unrestricted ability access enabled.

References and related help

Check the maintainer reference: AuraSkills documentation.

Before changing live data, read how to keep a completed recovery copy. For startup errors, use the log collection guide.

Keep AuraSkills reward commands from targeting the wrong player

A command reward must resolve to the player who earned the level, not the account testing the command. Compare substitution and execution context before connecting it to valuable rewards.

Check the installed content

AuraSkills connects skill XP sources to levels, rewards, abilities and menus. A change in one layer can leave another unchanged. Keep the existing skill identifiers and player data, and use the documentation matching your installed release when editing a source or reward.

Work through the change

1. Record the reward command and the intended recipient. Read the installed-release reward reference for supported player placeholders and whether the command runs as console or player.
2. Copy the reward to a staging level and replace valuable issuance with a harmless observable action supported by the target command. Keep command access narrowly scoped.
3. Trigger it with two distinct ordinary accounts in separate tests. Confirm that each result identifies the correct player and that the command does not contain a hard-coded staff or previous test name.
4. Restore the intended low-value reward and test one successful grant. Check that the target command receives one recipient and the planned quantity rather than accepting arbitrary player-controlled arguments.

Verify the result

Each account should receive only its own reward once at the intended transition. The final command should execute with the documented context and resolve its placeholders correctly.

If the result differs

If the placeholder remains literal, compare the supported syntax and the integration handling before adding aliases. If the command works manually but not as a reward, inspect console compatibility and execution errors.

Keep a recovery path

Restore the harmless test action if recipient resolution is uncertain. Keep valuable rewards disabled until both account tests pass and record any grant already issued so a retry does not create duplicate compensation.

References and related help

Check the maintainer reference: AuraSkills documentation.

Before changing live data, read how to keep a completed recovery copy. For startup errors, use the log collection guide.