PlotSquared tasks

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

- Allow visitors to use plot doors without granting building access
- Test a PlotSquared PvP arena without enabling combat on every plot
- Restrict a PlotSquared flag to the values a building role actually needs
- Distinguish a PlotSquared weather display from actual weather mechanics
- Give a PlotSquared build a fixed-looking time without stopping the world clock
- Limit plot entity counts without changing every world’s mob settings
- Disable redstone on one PlotSquared test plot and check its boundaries
- Check PlotSquared claim allowances against the permission-search range
- Verify WorldEdit cannot cross a PlotSquared owner boundary
- Add plot entry and exit messages without changing the world’s chat format

Allow visitors to use plot doors without granting building access

A public showroom needs usable entrances, but visitors rarely need permission to break walls or open every container. PlotSquared’s use flag accepts a material list. Build that list from the actual entrance blocks and keep the change local to the demonstration plot.

Commands beginning with a slash are player chat commands. Use them in game with the required role; check console support before adapting them for the hosting panel.

Work through the change

1. Identify the plot ID, owner and current use flag. Check inherited world defaults too; an absent local value can still produce an effective rule. Record both before editing.
2. Use `/plot flag info use` to confirm the installed flag format. Choose the precise door, button or gate materials needed for the entrance, rather than granting every interaction type.
3. As the authorised plot owner, apply the material list to the test plot. Use an ordinary visitor account to approach from the road and operate each entrance component.
4. Try a chest, an unrelated lever and a removable decoration as negative tests. Ask the owner to test normal building while the visitor remains present.

Check the result

The visitor should pass through the intended entrance while unrelated interactions and building remain restricted. Check both sides of double doors and any pressure plate used for exiting. A successful owner test cannot establish visitor permissions because ownership changes the protection result.

The use list concerns block materials rather than how important a particular door is. If the same permitted material appears elsewhere on the plot, include it in the visitor test. Choose the build’s

entrance hardware with that scope in mind before enabling the flag.

Recover or stop safely

Restore the recorded local flag or remove the override using the installed flag help. Do not replace world defaults to repair one doorway. If another protection plugin still denies access, investigate its matching region rule separately and keep plot protection enabled.

Related guidance and maintainer reference

For the broader setup, read Create a separate creative plot world with PlotSquared. Check the installed plugin version before applying syntax from the official PlotSquared documentation.

Test a PlotSquared PvP arena without enabling combat on every plot

Combat rules can come from PlotSquared, the server and another protection plugin. A plot-level PvP flag is one part of that chain. Test a single enclosed arena with willing participants before describing it as safe for public fights.

Commands beginning with a slash are player chat commands. Use them in game with the required role; check console support before adapting them for the hosting panel.

Work through the change

1. Record the arena plot ID and existing pvp and invincible flags. Note world-level PvP settings and any region plugin that also covers the site. Store valuables outside the test area.
2. Inspect `/plot flag info pvp` and apply the chosen boolean value only to the arena plot. Leave road and default world flags unchanged during this test.
3. Have two ordinary players test a single low-damage melee hit in the arena, then repeat on the neighbouring road and a control plot where combat should remain disabled.
4. Check a projectile separately and test entry and exit with combat equipment removed. Inspect death and inventory rules before conducting a lethal test; damage permission alone says nothing about dropped items.

Check the result

Combat should match the arena’s policy without changing the control plot. If damage remains blocked, compare invincibility and other protection layers rather than granting a global bypass. Record whether projectiles follow the same boundary so event organisers know the tested limits.

Check the arena boundary after participants leave combat mode as well as when they enter. Other plugins may retain combat tags or impose teleport rules. Those systems need their own

documented policy rather than a broader plot flag intended to compensate for an unexplained interaction.

Recover or stop safely

Restore the earlier arena flags and stop the trial if unexpected damage occurs outside it. Return test equipment from the staging chest. An arena rule does not reverse inventory loss, so use disposable equipment until death handling has been verified independently.

Related guidance and maintainer reference

For the broader setup, read Create a separate creative plot world with PlotSquared. Check the installed plugin version before applying syntax from the official PlotSquared documentation.

Restrict a PlotSquared flag to the values a building role actually needs

Flag administration can grant more authority than a building role requires. A player who needs to toggle one cosmetic option should not automatically gain access to plot combat, movement and interaction policies. Test permissions on an ordinary role with one specific flag.

Commands beginning with a slash are player chat commands. Use them in game with the required role; check console support before adapting them for the hosting panel.

Work through the change

1. Choose the exact flag and allowed values. Read its information with `/plot flag info` and record any basic flag-command permissions already inherited by the role.
2. Review the maintainer’s flag-specific permission schema for the installed release. Grant the required command permissions and the narrow permitted value nodes; avoid a wildcard that includes every flag or value.
3. Use a non-admin account that owns a disposable plot to set the approved value. Then attempt a different flag and an unapproved value so the denial path is tested too.
4. Reconnect the player and repeat the checks after permission caches settle. Inspect inherited groups if either denied operation succeeds, and compare the effective permission output rather than adding competing grants blindly.

Check the result

The player should manage the intended option while receiving a clear denial for unrelated flag changes. Verify that staff retain their separate administration path. A successful command as operator is not evidence that the limited building role has the correct permissions.

Keep the final permission nodes with the flag’s accepted values in the role documentation. A later plugin update may add options under a wildcard. Specific grants make the intended authority

easier to review without trying every new flag on a populated plot.

Recover or stop safely

Remove the trial grants and restore the test plot’s flags from your notes. If a broad inherited permission caused the leak, adjust the responsible group deliberately and retest affected roles. Do not solve a single missing value by giving the entire flag namespace.

Related guidance and maintainer reference

For the broader setup, read Create a separate creative plot world with PlotSquared. Check the installed plugin version before applying syntax from the official PlotSquared documentation.

Distinguish a PlotSquared weather display from actual weather mechanics

A plot can look sunny while the underlying world still follows its weather cycle. PlotSquared documents its weather flag as simulated weather, without changing weather-dependent mechanics. Check that distinction before using a cosmetic setting to protect a farm or a public event.

Commands beginning with a slash are player chat commands. Use them in game with the required role; check console support before adapting them for the hosting panel.

Work through the change

1. Choose a test plot beside an unchanged control plot. Record the current weather flag, world weather and any weather plugin that personalises what players see.
2. Read `/plot flag info weather` for supported values in your release. Apply a clear-weather display to the trial plot without changing the world’s weather cycle or another plot’s flags.
3. Walk across the plot boundary with two ordinary accounts. Compare the displayed rain and sky while also recording the world’s actual weather state through its existing administration controls.
4. Test the specific mechanic you care about separately, using expendable blocks or entities. Keep cosmetic appearance and game behaviour as different observations in the change notes.

Check the result

The selected weather appearance should apply inside the trial plot and end at the intended boundary. It should not be described as a world weather lock. Server plugins, client settings and version differences can alter what players see, so compare more than one client.

Tell event staff whether the setting is cosmetic or mechanical in the handover notes. That prevents a later organiser relying on a sunny-looking plot to suppress a weather-dependent hazard. Keep a screenshot of the control area and the actual world state with the trial result.

Recover or stop safely

Restore the earlier weather override or remove the local flag to return to inherited behaviour. If the aim was to change crop, mob or lightning behaviour, stop this cosmetic test and choose an appropriately scoped world or protection setting with its own recovery plan.

Related guidance and maintainer reference

For the broader setup, read Create a separate creative plot world with PlotSquared. Check the installed plugin version before applying syntax from the official PlotSquared documentation.

Give a PlotSquared build a fixed-looking time without stopping the world clock

Builders may want a permanent sunset for screenshots without freezing the whole world. PlotSquared’s time flag changes the simulated time seen on the plot. It does not change time-dependent game mechanics, so document that limitation before using it for a public build.

Commands beginning with a slash are player chat commands. Use them in game with the required role; check console support before adapting them for the hosting panel.

Work through the change

1. Record the plot ID, local time flag and any inherited default. Pick a neighbouring plot as a control and note whether a personal-time plugin is already active.
2. Inspect `/plot flag info time` for the installed release’s accepted format. Apply the chosen display time only to the trial plot, preserving the world’s ordinary daylight-cycle setting.
3. Enter and leave the plot with an ordinary visitor. Compare the sky at the same real moment with another player standing in the control area, then reconnect and check again.
4. If the build includes daylight sensors or sleeping behaviour, test those separately. Record their actual response rather than assuming a visible sunset means the server clock is also fixed.

Check the result

The visual time should match the selected value inside the plot while the world continues its normal cycle. Test any resource pack or shader used for screenshots because it can change perceived brightness. Visitors should regain the expected display after leaving the area.

If you publish a screenshot location, include the selected display time in its build notes. Another player may otherwise change the world clock trying to reproduce the same image. A local visual

setting keeps that artistic choice separate from the rest of the server’s schedule.

Recover or stop safely

Restore the original flag or remove the override to inherit the world default. If the display remains fixed elsewhere, check personal-time permissions and other plugins before altering the global clock. Keep a screenshot of the control area to distinguish a client display issue from world state.

Related guidance and maintainer reference

For the broader setup, read Create a separate creative plot world with PlotSquared. Check the installed plugin version before applying syntax from the official PlotSquared documentation.

Limit plot entity counts without changing every world’s mob settings

A crowded creative plot can accumulate more entities than its owner intended. PlotSquared exposes separate entity-cap flags, including animal, hostile and vehicle categories. Choose the category that matches the actual load rather than applying an unexplained global mob restriction.

Commands beginning with a slash are player chat commands. Use them in game with the required role; check console support before adapting them for the hosting panel.

Work through the change

1. Inspect the affected plot and list its entities by purpose: displays, pets, villagers and vehicles. Record the existing cap flags and any other entity-management plugins before changing admission rules.
2. Use `/plot flag info` for the specific cap you need. Read the maintainer’s entity categories because a familiar creature or object may fall under more than one relevant group.
3. Apply a conservative trial cap on an empty demonstration plot. Add expendable examples of the target category and check what happens at the boundary without testing on valuable named pets.
4. Check a different category and a neighbouring plot as controls. Observe both player-created entities and any natural spawning relevant to the intended policy, then document the behaviour of your installed build.

Check the result

The trial should constrain the intended category while preserving unrelated plot behaviour. Count actual entities rather than relying only on visible mobs, particularly for small decorative objects. A cap does not establish that a plugin will clean up existing entities in the way you expect.

Choose an operational policy for existing over-cap plots before rolling out the limit. An admission restriction and an automatic cleanup are different behaviours. Owners should know how their retained decorations or animals are handled rather than discovering the policy through a failed interaction.

Recover or stop safely

Restore the original cap if the trial affects the wrong category. Do not clear a live plot to force compliance with a new limit. Agree a separate preservation or cleanup plan with the owner before changing valuable entities already stored in their build.

Related guidance and maintainer reference

For the broader setup, read Create a separate creative plot world with PlotSquared. Check the installed plugin version before applying syntax from the official PlotSquared documentation.

Disable redstone on one PlotSquared test plot and check its boundaries

An event plot may need redstone disabled while nearby building plots continue normally. A local PlotSquared flag can scope that policy, but existing clocks and devices deserve careful testing. Start with a simple circuit instead of a finished storage system.

Commands beginning with a slash are player chat commands. Use them in game with the required role; check console support before adapting them for the hosting panel.

Work through the change

1. Identify the target plot and record its redstone flag, inherited defaults and nearby merged plots. Ask the owner to pause active machinery and save the current server state before testing.
2. Create a small expendable lever-and-lamp circuit wholly inside the plot and a second circuit on a control plot. Record the initial states so delayed updates are easier to distinguish.
3. Inspect `/plot flag info redstone` and apply the local disabled setting. Toggle both trial circuits, observing the target and control area separately without altering road defaults.
4. Test boundary-adjacent components within permitted build space. Review any redstone optimisation plugin if the result differs from the flag description, and keep high-value machinery out of the trial.

Check the result

The intended plot should follow the local redstone policy while the control circuit retains its previous behaviour. Existing powered states can make a single screenshot misleading; repeat the interaction after the normal update interval and record the actual device response.

Keep machinery paused until the local test is complete. Restoring redstone permission can trigger queued or newly permitted updates in a complex build. Have the owner inspect item paths and storage capacity before returning a large machine to normal unattended operation.

Recover or stop safely

Restore the earlier flag and verify the small circuit responds again before restarting larger machinery. If a device remains stuck, diagnose that component’s saved state with its owner. Do not remove blocks from a working storage machine to demonstrate that protection has changed.

Related guidance and maintainer reference

For the broader setup, read Create a separate creative plot world with PlotSquared. Check the installed plugin version before applying syntax from the official PlotSquared documentation.

Check PlotSquared claim allowances against the permission-search range

PlotSquared determines claim allowances from permissions. Its configured limit.max-plots value controls the integer-permission range searched; it does not itself grant players that number of plots. Diagnose the effective allowance and lookup range separately, using a trial account rather than changing live ownership.

Work through the change

1. Record the player’s current claimed plots and relevant effective claim permissions. Note group inheritance and whether the account is operator or has a bypass role.
2. Read the installed PlotSquared permissions documentation and inspect limit.max-plots in the generated settings. The maintainer defines it as the maximum integer-permission range to check. Raising it alone does not change a player’s allowance; keep the configuration and permission export together.
3. Use an ordinary test account with a small known allowance in a disposable plot area. Claim up to that allowance and attempt one extra claim, recording the response and plot count.
4. Set the intended account allowance through the documented claim-limit permission. Adjust the search range only if it would prevent that permission from being considered. Repeat the trial after the supported permission refresh or controlled configuration restart.

Check the result

The trial account should claim its permission-derived number of plots and receive a denial for the next one. Existing ownership should remain untouched. If the result differs, inspect effective permissions and per-area behaviour; changing the global search range is not a replacement for granting the intended allowance.

Keep a sample permission-derived allowance for each paid rank in the test record. That catches inheritance mistakes while retaining the shared permission-search range as a separate setting. An increase for one customer should change that role’s intended grant, not rely on changing a global lookup setting.

Recover or stop safely

Restore the trial allowance and release only the disposable test claims using the installed owner controls after checking they contain no builds. Keep live player claims intact. A commercial downgrade needs a separate policy for excess plots rather than silently deleting them.

Related guidance and maintainer reference

For the broader setup, read Create a separate creative plot world with PlotSquared. Check the installed plugin version before applying syntax from the official PlotSquared documentation.

The official PlotSquared settings source explains the permission-search limit; compare it with your installed release and generated configuration.

Verify WorldEdit cannot cross a PlotSquared owner boundary

Plot protection and WorldEdit must agree before builders edit near a neighbour. PlotSquared provides integration features, while broad editing bypass permissions can defeat the intended boundary. Use a marked test strip and a normal builder account to verify the combined result.

Work through the change

1. Prepare two adjacent test plots with different owners and expendable blocks along their edges. Record the builder’s WorldEdit, FAWE and PlotSquared bypass permissions before making any edit.
2. Confirm the installed WorldEdit or FAWE integration matches the supported versions. Check whether the plot has a no-worldedit flag and distinguish a complete editing denial from a boundary restriction.
3. With the ordinary builder, select a tiny area spanning both plots and perform one reversible material change. Compare the report with the blocks on each owner’s side.
4. Repeat at a road edge and on the builder’s own interior. Restore the test blocks between trials so a previous selection or mask cannot hide the effect of protection.

Check the result

Edits should affect only the area authorised for that account, or fail according to the installed integration’s documented policy. A successful administrator test proves little because bypass roles differ. Keep screenshots and coordinates of each control block with the permission snapshot.

Preserve a screenshot of the selection and the resulting edge blocks. A command rejection and a partial edit are different outcomes that staff should recognise. Repeat the small trial after major integration updates before allowing routine editing beside valuable neighbouring builds.

Recover or stop safely

Undo the trial immediately and check both plot edges. Remove any temporary bypass grant used for diagnosis. If integration is missing or incompatible, disable ordinary access to editing commands until a supported combination is installed; do not rely on verbal boundary instructions.

Related guidance and maintainer reference

For the broader setup, read Create a separate creative plot world with PlotSquared. Check the installed plugin version before applying syntax from the official PlotSquared documentation.

Add plot entry and exit messages without changing the world’s chat format

A plot greeting can tell visitors where they are without replacing the server’s chat plugin. Keep the message short, local and free of account or connection details. Test repeated entry so an attractive signpost does not become constant chat noise.

Commands beginning with a slash are player chat commands. Use them in game with the required role; check console support before adapting them for the hosting panel.

Work through the change

1. Record the plot’s current greeting and farewell values, including whether each is inherited from world defaults. Choose a short location description rather than promotional links or commands.
2. Check `/plot flag info greeting` and farewell for your installed release. Apply the greeting only to the trial plot and test its accepted text format before adding elaborate formatting.
3. Enter from several directions with an ordinary visitor and leave again. Note whether the message appears in chat and whether repeated boundary crossings produce excessive notices.
4. Compare a neighbouring plot and normal chat messages. If another plugin also sends region greetings, decide which system should own this particular location before enabling both publicly.

Check the result

The visitor should receive the correct local message without changes to global chat prefixes or unrelated plots. Check colour readability with the server’s normal resource pack and chat settings. A greeting should identify the place clearly even when formatting is unavailable.

Choose a message that still makes sense without colour formatting or clickable text. Visitors may use different chat settings and clients. If a boundary is crossed frequently, prefer one useful location name over a long paragraph that hides normal conversation and event instructions.

Recover or stop safely

Restore the prior text or remove the local overrides to inherit defaults. If duplicate messages remain, identify the second region or greeting plugin rather than changing global chat format. Keep the original text in your notes so punctuation and links can be restored exactly.

Related guidance and maintainer reference

For the broader setup, read Create a separate creative plot world with PlotSquared. Check the installed plugin version before applying syntax from the official PlotSquared documentation.