Request rules: decide what happens to a request before its view runs

This page is for administrators who control which addresses visitors may use, for example to hide system URLs such as /content/view/full/123. A URL alias becomes content/view/full/ once the kernel has translated it, so…

Source file: doc/eng-us/features/6.0/request-rules.md Last changed 2026-11-04
https://doc.exponential.earth/learninminutes/exphelp/viewer/features/6.0/request-rules.md

This page is for administrators who control which addresses visitors may use, for example to hide system URLs such as `/content/view/full/123`. A URL alias becomes `content/view/full/<node>` once the kernel has translated it, so neither `[SiteAccessRules]` nor `content/read` limitations could tell `/content/view/full/123` from the same page at its alias. Request rules close that gap. They run after the URL alias is translated and after the role policies allowed the view, and can allow, deny, redirect, rewrite or log. Added 2026-10-02.

Complete guide with every condition, action, fact and sixteen recipes: View full security.

Quick start: send system URLs to their alias

1. See who would be affected first. Enable the audit rule, which only writes a line to `var/log/requestrules.log` for each system URL typed by a user without `content/view_system_url` and changes nothing.
2. Then enable the redirect. In `settings/siteaccess/<site>/requestrules.ini.append.php` (or `settings/override/`):

INI

```
<?php /*
[RequestRuleSettings]
RuleList[]=system_url_full_view_to_alias
*/ ?>
```

3. Check what the rules decide for any address, as any user, without changing anything:

Shell

```
php bin/php/ezrequestrules.php -s site --check
php bin/php/ezrequestrules.php -s site --uri=content/view/full/2
```

A visitor who types `/content/view/full/123` is now redirected (301) to the node's alias, unless they hold the new policy `content/view_system_url`. Without an alias, the answer is "not found".

Settings

File	Block	Key	Default	Meaning

File	Block	Key	Default	Meaning
settings/requestrules.ini	RequestRuleSettings	Enabled	true	With false no rule is asked; the kernel behaves as before
settings/requestrules.ini	RequestRuleSettings	RuleList[]	empty	Rules asked in this order; the first whose conditions all match decides
settings/requestrules.ini	RequestRuleSettings	RuleProviders[]	empty	Classes implementing ezpRequestRuleProvider (rules computed in PHP)
settings/requestrules.ini	RequestRuleSettings	FactProviders[]	empty	Classes implementing ezpRequestFactProvider, for Conditions[fact:<name>]
settings/requestrules.ini	RequestRuleSettings	MaxRewrites	3	Rewrites per request before "not found" (stops loops)
settings/requestrules.ini	RequestRuleSettings	ConditionHandlers[], ActionHandlers[]	empty	Extension conditions and actions by name
settings/requestrules.ini	Rule-<name>	Conditions[...], Action, ActionArgs[...], FallbackAction, Description	per rule	One rule

Scope: override or siteaccess. A siteaccess override that starts with an empty `RuleList[]` replaces the list.

Check the shipped defaults (`Enabled=true`, an empty `RuleList[]`, `MaxRewrites=3`):

Shell

```
grep -n "RuleList\|Enabled\|MaxRewrites" settings/requestrules.ini
```

What a rule can say

- **25 conditions**: `requested_via` (system, alias, wildcard, front page), `uri`, `module_view`, `view_mode`, `node`, `subtree`, `class`, `section`, `state`, `user`, `role`, `policy`, `siteaccess`, `host`, `header`, `ip`, `fact`, and more.
- **8 actions**: `allow`, `notfound`, `forbidden`, `login`, `redirect`, `redirect_to_alias`, `rewrite`, `log`.

Three rules ship, all off:

Rule	What it does
system_url_full_view_to_alias	Redirects a typed system full view URL to the alias

Rule	What it does
`system_url_content_view_notfound`	Answers every system content view with "not found"
`system_url_full_view_audit`	Writes a line to `var/log/requestrules.log` for each system URL typed by a user without `content/view_system_url`; changes nothing. Put it first to see who would be affected before enforcing a stricter rule

Safety

- Rules only take access away or send the visitor elsewhere; they never give access a policy refused.
- Any answer except `allow` is sent private, no-store. A page that a rule with a request-dependent condition (header, ip, user) could decide stays out of the role-aware HTTP cache and the Velocity response cache; pages the rules cannot affect are cached as before.
- The `login` action never acts on the sign-in views, so no rule can loop.
- Rules read facts only, resolved on first use, so they are tested without a database: 21 unit tests in `tests/tests/kernel/classes/requestrules/ezpRequestRuleEngineTest.php`. The 25 conditions and 8 actions are the arrays `$builtInConditions` and `$builtInActions` of `ezpRequestRuleEngine`.

Related pages

- Hardening
- The audit trail
- Object states: the states `Conditions[state]` matches
- Admin links follow permissions
- Upgrade checklist of 1-2 October 2026
- 6.0.15 changelog
- October 2026 chronicle

> **Become an Exponential Partner**
>
> Partners share their information, provide funding, contribute back and develop the products with us, and that is how Exponential grows. A partner website and dashboard are planned for mid-2027: please become a partner today to speed up our growth rate and land new features faster.
>
> Partners: start here | How to apply

Exponential Copyright © 1998-2026 Exponential Foundation and 7x
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".
This documentation is free documentation, licensed under the GNU Free Documentation License 1.3 or later:
https://www.gnu.org/licenses/fdl-1.3.html.