

# How does Exalate handle sync failures?

Last Modified on 02/17/2026 9:44 am EST

When a synchronization fails, **Exalate logs the error** and displays it in the unified console dashboard, offering detailed insights into what went wrong. This helps you quickly identify the issue and fix it without needing to sift through system logs.

## Error logging and visibility in the unified console

The **error view** in the dashboard provides:

- The **connection** where the failure occurred
- The **specific work item** involved
- The **error message** and **line in your script** where the problem happened

This visibility makes troubleshooting more straightforward and saves time in resolving sync failures.

## Queuing failed synchronization attempts

Failed sync attempts are not lost—they remain **queued** in the system until the underlying issue is fixed. The affected work items stay in a **pending state**, ensuring no data is lost. Once the cause of the failure (like a script error or connectivity problem) is resolved, you can **retry synchronization** for the affected items.

This queuing mechanism ensures **no data loss**, even during temporary problems preventing successful sync.

## Aida's assistance for troubleshooting sync failures

In the **New Exalate experience**, **Aida** provides **context-aware assistance** for troubleshooting sync failures. When an error occurs, Aida helps by:

- Explaining the problem in an accessible language
- Suggesting **potential solutions** based on the error type and your configuration

For example, if a script tries to access a property that doesn't exist, Aida identifies the missing field and guides you on how to handle it properly.

Workspaces / Orlando

Orlando

Integrates with ServiceNow to provide real-time updates on incident status, errors, and deployment progress, enhancing support workflows.

Connections

Systems

Troubleshooting

Errors in Orlando

Search

Occurred	Error			
Just now	<b>Outgoing script error</b> Cannot get property 'value' on null object <b>Error line: 18</b>	Dev Team / DevTeam_to_Support	Conenction	Resolve <div></div>
25 minutes ago	<b>Script execution failure</b> Cannot get property 'value' on null object	Support / Support_to_CloudOps	Item	Resolve <div></div>
1 hour ago	<b>TrackerResetException</b> Cannot get property 'value' on null object	CloudOps / DevTeam_to_CloudOps	System	Resolve <div></div>

Aida Diagnosis

The error message you're seeing indicates a script-related issue: Cannot get property 'value' on null object. This means there's likely an attempt in your script to access a property on an object that doesn't exist or hasn't been initialized....

View Full Analysis >

John Doe

System Admin

Logout

Workspaces / Orlando

Outgoing script error

Just now

Resolve

Aida Diagnosis

Error Details

Replicas

Copy

The error "Cannot get property 'value' on null object" in your outgoing sync script means that the script is trying to access a property (like .value) on an object that doesn't exist (it's null). This typically happens when a custom field or property you're referencing isn't present on the issue being synced.

Here's how to fix it:

1. Check your outgoing sync script, especially around line 18 (as mentioned in the stack trace: Script13.groovy:18).
2. Look for any line trying to access .value on a field, such as issue.customFields."SomeField".value.
3. The field "SomeField" may not exist on the issue, meaning issue.customFields."SomeField" is null.

To prevent this error, use Groovy's null-safe operator (?.) like this:

```
def fieldValue = issue.customFields."SomeField"?.value
```

This ensures that if the field doesn't exist, it won't throw an error.

After updating your script, retry the synchronization. If you want to debug further, you can add a line like:

```
debug.error("Field value is: " + issue.customFields."SomeField")
```

This will help you see what value (if any) is being passed.

Let me know if you need help finding the exact line or want to share your script for a quick review!

## Common causes of sync failures

Sync failures can be caused by several factors:

- **Script errors**, such as **null pointer exceptions** or **syntax mistakes**
- **Permission issues**, where Exalate lacks access to the required fields

- **Network connectivity problems**
- **Validation failures**, where incoming data doesn't meet the receiving system's requirements

The error messages provide insight into which category the failure falls under, helping you target your troubleshooting efforts effectively.

## Ensuring data integrity during failures

Exalate ensures that other work items continue to sync successfully, even if specific items fail.

**One problematic work item** (due to a script error or other issue) **won't block** the entire connection from functioning, which means most of your integration continues operating while you address the failure.

## Testing fixes with draft versions and Test Run

For persistent errors that affect multiple items, you can create a **draft version** of your configuration and test fixes using the **Test Run** feature. This allows you to try changes in a safe environment and prevent further failures while you work on a solution.

This approach to handling sync failures ensures **continuous synchronization**, effective troubleshooting, and **data integrity** during issues. With Exalate's built-in features, you can quickly resolve issues and keep the integration running smoothly.

The screenshot displays the Exalate configuration interface. At the top, a green banner states: "You're in draft mode now. You can edit and save and /publish this version." Below this, the breadcrumb navigation shows "Workspaces / Orlando / Support Team to Dev Team". The main heading is "Connection: Support Team to Dev Team". A status bar indicates "Version 3 (Draft)" with a "Publish version" button. To the right is an "Item Sync Monitor" link.

The "Triggers" section contains a table with the following data:

Item type	Query	Status	Actions
Issue	Project = EXA	On (Green toggle)	Edit, More options
Sprint	Project = SUP	On (Green toggle)	Edit, More options

A blue arrow points from the "Start test run" button in the "Scripts" section to the "Status" column of the "Triggers" table. The "Scripts" section is divided into "Outgoing script" (From Dev Team) and "Incoming script" (Into Support Team). The "Incoming script" contains the following code:

```

1 if(firstSync){
2   issue.projectKey = nodeHelper.getProjectByKey("DT")
3   // Set this name from source issue. If not found set a default

```

The left sidebar contains navigation links for "Workspaces", "Users", "Settings", "Resources", "Product", "About Us", "Release History", "Glossary", "API Reference", "Security", "Pricing and Licensing", "Resources", "Subscribe for a weekly Exalate hack", "Academy", and "Blog". The user "John Doe" is logged in as "System Admin".

[YouTube Channel](#) 

[Ebooks](#) 

**Still need help?**

[Join our Community](#) 

[Visit our Service Desk](#) 

[Find a Partner](#) 