Debugging Sync scripts on Exalate

Last Modified on 03/04/2024 7:56 am EST

Sometimes you need to have a closer look at synchronization scripts to find out what's wrong.

There are two ways to get feedback from the Exalate App:

- Exalate logging: it doesn't block the synchronization and requires the processor changes to run again. We discuss this in the next section.
- Debugging error from the scripts - blocks the entity synchronization and requires ***Resolve and retry*** to run again.

Debugging Errors from the Scripts

To get more details about values passed in a replica you can use **debug.error** method.

debug.error method

This method helps to raise an IssueTrackerException error with the value of the field which is not syncing.

It allows you to see the content of the replica and the issue (if applicable). This helps to get a better understanding of what stage the problem has occurred.

```
debug.error("String message")
```

For example, when you synchronize a custom field, but the issue on the receiving side is not updated and you don't see any error.

The script below shows how to throw an error with the custom field value stored in a replica.

Receiving side - Incoming sync

```
debug.error("customFields."Foo".value from remote side is: " + replica.customFields."Foo"?.value)
```

How to Get Only Debug Logging for the Scripts?

If you want to debug your scripts, then capturing all **com.exalate** logging is too much as it also captures all transport protocol details.

You can limit the logging to different levels:

- for only script logging use **com.exalate.script**
- for information on how scripts run use **com.exalate.processor**

For Exalate 4.2 and lower

- for scripts and information on how they run use **com.exalate.processor**

- for Outgoing sync(data filter) scripts
  use **com.exalate.processor.jira.JiraCreateReplicaProcessor**
- for Incoming sync for new issues(create processor) scripts
  use **com.exalate.processor.jira.JiraCreateIssueProcessor**
- for Incoming sync for existing(change processor) scripts use
  com.exalate.processor.jira.JiraChangeIssueProcessor

You can test how logging works by adding the code, provided below. It will add the 'Hello world' text to the log file.

```
//add a "Hello world" to the logging
log.info("Hello world ...")
```