How to Sync Attachments in Azure DevOps

Last Modified on 04/03/2024 7:38 am EDT

This article describes how to synchronize attachments.

With Exalate, you have full control over attachment synchronization. You can define the sync behavior for attachments in different ways. For example:

- filter what attachments to send by file type or size
- sync attachments from the public comments only

The maximum attachment size that you can synchronize depends on a couple of factors:

- the attachment size limit on both instances (you can configure the max attachment size in Jira)
- instance HTTP/HTTPS connection timeouts

We've tested the synchronization of attachments with the size up to 1 GB, which worked without any issues, bigger attachments should not be a problem either.

Source Side

When sending attachments, you need to remember, that not all the attachments have to be sent over. You can use the groovy collection methods to control the collection contents:

Outgoing sync

- Send all the attachments

```
replica.attachments    = issue.attachments
```

- Send only *.pdf* files

```
replica.attachments = issue.attachments.findAll { attachment -> attachment.filename.endsWith(".pdf") }
```

- Send only attachments mentioned in public comments and handle names with special characters

```
def isMentionedInPublicComments(attachment, publicComments) {
    publicComments?.any {c ->
        c.body.contains("[^" + attachment.filename + "]") ||
       (c.body =~ (/!${java.util.regex.Pattern.quote(a.filename)}(\|.*!|!)/.toString())).find()
    }
}
replica.attachments = issue.attachments.findAll{isMentionedInPublicComments(it, publicComments)}
```

Destination Side

Incoming sync

You can define how to create issue attachments, received from the other side.

Check some examples below:

- Merge attachments using attachmentHelper.mergeAttachments . It's is the most common case, which is included in the default sync rules.

```
//add all new attachments as listed in the replica and remove all attachments from the issue which have been remove
d in the remote issue
issue.attachments = attachmentHelper.mergeAttachments(issue, replica)
```

- Create new attachments, received from the source side

```
 issue.attachments.addAll(replica.addedAttachments)
// or  issue.attachments += replica.addedAttachments
```

- Add all received attachments to the local issue and change the attachment name. You can manipulate attachments via groovy collection methods to apply any specific behavior.

```
issue.attachments.addAll(replica.addedAttachments)
issue.attachments = issue.attachments.collect {attachment ->
        attachment.filename = "node_A_" + attachment.filename
        // return an attachment
       attachment
}
```

- Gather statistics from the attachments

```
def numberOfAttachmentsPerTypes = issue.attachments.inject([:]) { result, attachment ->
 def getFileExt = {
    filename -> def lastDotIdx = filename.lastIndexOf("."); (lastDotIdx > 0)? filename.substring(lastDotIdx) : null
   }
 def fileExt = getFileExt(attachment.filename)
 def numberOfAttachmentsPerType = result[fileExt]
 numberOfAttachmentsPerType = numberOfAttachmentsPerType ?: 0
 result[fileExt] = numberOfCommentsPerAuthor + 1
 result
}
/*
for attachments:[
 [
  filename:"foo.pdf"
 ],
 [
  filename:"bar.txt"
 ],
 [
  filename:"baz.pdf"
 ],
 [
  filename:"hahaha"
 ]
]
 numberOfAttachmentsPerTypes would be [".pdf" : 2,".txt" : 1, null : 1]
*/
```

- Make the issue attachments the same as the ones received from the remote instance

This removes all the attachments, which are not present on the remote issue replica. Use with caution.

```
issue.attachments = replica.attachments
```

Have more questions? Ask the community

Product
About Us
Release History
Glossary
API Reference
Security
Pricing and Licensing
Resources
Academy
Blog
YouTube Channel
Ebooks
Still need help?
Join our Community
Visit our Service Desk
Find a Partner