Base64 Attachment Upload REST API

Atlassian Marketplace Plugin Documentation

CONTENT

This plugin decodes Base64 encoded attachments using provided REST API

/1 COMMUNICATION

There are two options to communicate with plugin.

1. **Directly** - Allows direct upload of base64 encoded attachments where metadata are stored in URL
2. **Json** - This option provides endpoint for metadata being stored in JSON structure

/1.1 COMMUNICATION ENDPOINTS

To decode base64 attachments you have to send request using one of the provided endpoints. There are two endpoints available. One for each communication option.

- **Direct endpoint**

Endpoint for direct communication

```
https://{JiraURL}/rest/base64attachment/1.0/direct/{issueIdOrKey}/{filename}/
```

- **JSON endpoint**

Endpoint for direct communication

```
https://{JiraURL}/rest/base64attachment/1.0/json
```

/2 AUTHENTICATION

Both direct and JSON options require preemptive basic authentication.

/3 DIRECT REQUEST

Allows direct upload of base64 encoded attachment where metadata are in URL.

/3.1 URL

URL should consist of your Jira instance address followed by direct endpoint.

The URL can be parametrized as follows:

- {**issueIdOrKey**} - path parameter that should be replaced with the issue id or key
- {**filename**} - path parameter that should be replaced with filename that will be displayed in Jira

Additionally, following optional query parameters may be used:

- {**contentType**} - if used, then Jira will store this content type for the uploaded attachment. If not specified, then internal Jira functionality will decide best content type for uploaded attachment.
- {**zipFlag**} - if used, then the zip flag will be passed to Jira to indicate that the uploaded attachment is a zip archive.

/3.2 METHOD

HTML Post Method

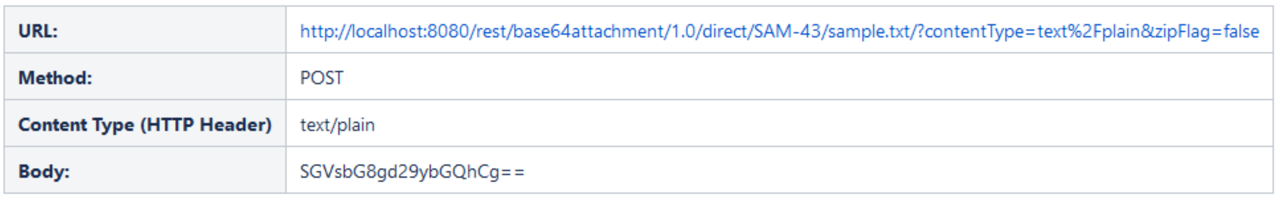
/3.3 CONTENT TYPE (HTTP HEADER)

text/plain

/3.4 BODY

Request body should contain base64 encoded attachment data.

/3.5 EXAMPLE OF DIRECT REQUEST



URL:	http://localhost:8080/rest/base64attachment/1.0/direct/SAM-43/sample.txt/?contentType=text%2Fplain&zipFlag=false
Method:	POST
Content Type (HTTP Header)	text/plain
Body:	SGVsbG8gd29ybGQhCg==

/4 JSON REQUEST

Allows upload of base64 encoded attachment specified as a JSON structure.

/4.1 URL

URL should consist of your Jira instance address followed by direct endpoint.

/4.2 METHOD

HTML Post Method

/4.3 CONTENT TYPE (HTTP HEADER)

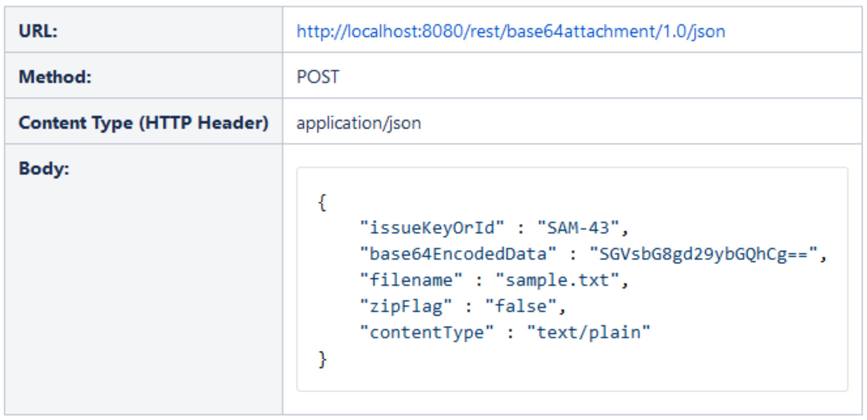
text/plain

/4.4 BODY

The JSON structure included in body can be parametrized as follows:

- **issueIdOrKey** - required parameter, should be replaced with the issue id or key
- **base64EncodedData** - required parameter, should contain base64 encoded attachment data
- **filename** - required parameter, should be replaced with filename that will be displayed in Jira
- **contentType** - optional parameter, if used then Jira will store this content type for the uploaded attachment (If not specified, then internal Jira functionality will decide best content type for uploaded attachment)
- **zipFlag** - optional parameter, if used then the zip flag will be passed to Jira to indicate that the uploaded attachment is a zip archive

/4.5 EXAMPLE:



URL:	http://localhost:8080/rest/base64attachment/1.0/json
Method:	POST
Content Type (HTTP Header)	application/json
Body:	

```
{
    "issueKeyOrId" : "SAM-43",
    "base64EncodedData" : "SGVsbG8gd29ybGQhCg==",
    "filename" : "sample.txt",
    "zipFlag" : "false",
    "contentType" : "text/plain"
}
```

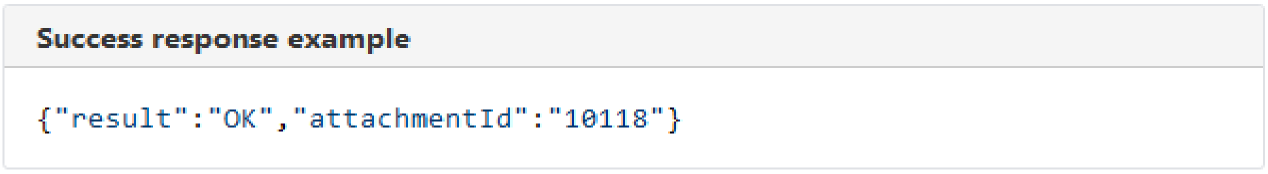
/5 RESPONSE

/5.1 SUCCESS

Both endpoints respond with a JSON response that contain following fields:

- **result** - OK, when attachment was correctly created
- **attachmentId** - attachment ID

/5.1.1 SUCCESS RESPONSE EXAMPLE


Success response example

```
{"result":"OK","attachmentId":"10118"}
```

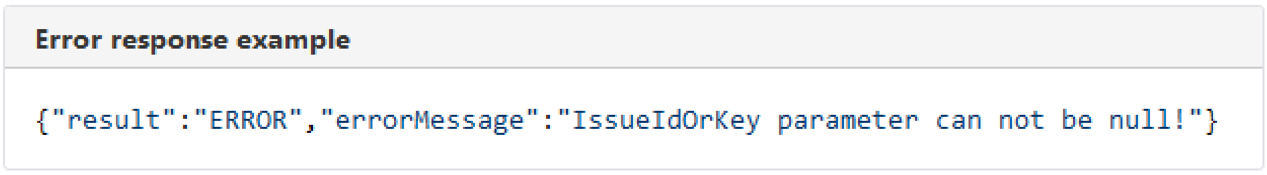
/5.2 ERROR

Both endpoints respond usually with a JSON response that will contain following fields:

- **result** - ERROR
- **errorMessage** - error message

The HTTP status code of an error response will be in the 4** or 5** range.

/5.2.1 ERROR RESPONSE EXAMPLE:


Error response example

```
{"result":"ERROR","errorMessage":"IssueIdOrKey parameter can not be null!"}
```