The REST API is now versioned. For more information, see "About API versioning."

Git tags

Use the REST API to interact with tag objects in your Git database on GitHub Enterprise Server.

About Git tags 🔗

A Git tag is similar to a Git reference, but the Git commit that it points to never changes. Git tags are helpful when you want to point to specific releases. These endpoints allow you to read and write tag objects to your Git database on GitHub Enterprise Server. The API only supports annotated tag objects, not lightweight tags.

Create a tag object 🔗

✓ Works with GitHub Apps

Note that creating a tag object does not create the reference that makes a tag in Git. If you want to create an annotated tag in Git, you have to do this call to create the tag object, and then create the `refs/tags/[tag]` reference. If you want to create a lightweight tag, you only have to create the tag reference - this call would be unnecessary.

Signature verification object

The response will include a `verification` object that describes the result of verifying the commit's signature. The following fields are included in the `verification` object:

Name	Type	Description
`verified`	boolean	Indicates whether GitHub considers the signature in this commit to be verified.
`reason`	string	The reason for verified value. Possible values and their meanings are enumerated in table below.
`signature`	string	The signature that was extracted from the commit.
`payload`	string	The value that was signed.

These are the possible values for `reason` in the `verification` object:

Value	Description
`expired_key`	The key that made the signature is expired.
`not_signing_key`	The "signing" flag is not among the usage flags in the GPG key that made the signature.

`gpgverify_error`	There was an error communicating with the signature verification service.
`gpgverify_unavailable`	The signature verification service is currently unavailable.
`unsigned`	The object does not include a signature.
`unknown_signature_type`	A non-PGP signature was found in the commit.
`no_user`	No user was associated with the `committer` email address in the commit.
`unverified_email`	The `committer` email address in the commit was associated with a user, but the email address is not verified on their account.
`bad_email`	The `committer` email address in the commit is not included in the identities of the PGP key that made the signature.
`unknown_key`	The key that made the signature has not been registered with any user's account.
`malformed_signature`	There was an error parsing the signature.
`invalid`	The signature could not be cryptographically verified using the key whose key-id was found in the signature.
`valid`	None of the above errors applied, so the signature is considered to be verified.

Parameters for "Create a tag object"

Headers

accept   string

Setting to `application/vnd.github+json` is recommended.

Path parameters

owner   string   Required

The account owner of the repository. The name is not case sensitive.

repo   string   Required

The name of the repository without the `.git` extension. The name is not case sensitive.

Body parameters

tag   string   Required

The tag's name. This is typically a version (e.g., "v0.0.1").

message   string   Required

The tag message.

object   string   Required

The SHA of the git object this is tagging.

The SHA of the git object this is tagging.

---

type string Required

The type of the object we're tagging. Normally this is a `commit` but it can also be a `tree` or a `blob`.

Can be one of: `commit`, `tree`, `blob`

---

tagger object

An object with information about the individual creating the tag.

▶ Properties of `tagger`

HTTP response status codes for "Create a tag object"

Status code	Description
201	Created
422	Validation failed, or the endpoint has been spammed.

Code samples for "Create a tag object"

POST /repos/{owner}/{repo}/git/tags

cURL   JavaScript

```
curl -L \ -X POST \ -H "Accept: application/vnd.github+json" \ -H "Authorization: Bearer <YOUR-TOKEN>" \ -H "X-GitHub-Api-Version:
2022-11-28" \ http(s)://HOSTNAME/api/v3/repos/OWNER/REPO/git/tags \ -d '{"tag":"v0.0.1","message":"initial
version","object":"c3d0be41ecbe669545ee3e94d31ed9a4bc91ee3c","type":"commit","tagger":{"name":"Monalisa
Octocat","email":"octocat@github.com","date":"2011-06-17T14:53:35-07:00"}}'
```

Response

Example response   Response schema

Status: 201

```
{ "node_id": "MDM6VGFnOTQwYmQzMzYyNDhlZmFlMGY5ZWU1YmM3YjJkNWM5ODU4ODdiMTZhYw==", "tag": "v0.0.1", "sha":
"940bd336248efae0f9ee5bc7b2d5c985887b16ac", "url": "https://HOSTNAME/repos/octocat/Hello-
World/git/tags/940bd336248efae0f9ee5bc7b2d5c985887b16ac", "message": "initial version", "tagger": { "name": "Monalisa Octocat",
"email": "octocat@github.com", "date": "2014-11-07T22:01:45Z" }, "object": { "type": "commit", "sha":
"c3d0be41ecbe669545ee3e94d31ed9a4bc91ee3c", "url": "https://HOSTNAME/repos/octocat/Hello-
World/git/commits/c3d0be41ecbe669545ee3e94d31ed9a4bc91ee3c" }, "verification": { "verified": false, "reason": "unsigned",
```

Get a tag 🔗

✅ Works with GitHub Apps

Signature verification object

The response will include a `verification` object that describes the result of verifying the commit's signature. The following fields are included in the `verification` object:

Name	Type	Description

verified	boolean	Indicates whether GitHub considers the signature in this commit to be verified.
reason	string	The reason for verified value. Possible values and their meanings are enumerated in table below.
signature	string	The signature that was extracted from the commit.
payload	string	The value that was signed.

These are the possible values for `reason` in the `verification` object:

Value	Description
expired_key	The key that made the signature is expired.
not_signing_key	The "signing" flag is not among the usage flags in the GPG key that made the signature.
gpgverify_error	There was an error communicating with the signature verification service.
gpgverify_unavailable	The signature verification service is currently unavailable.
unsigned	The object does not include a signature.
unknown_signature_type	A non-PGP signature was found in the commit.
no_user	No user was associated with the `committer` email address in the commit.
unverified_email	The `committer` email address in the commit was associated with a user, but the email address is not verified on their account.
bad_email	The `committer` email address in the commit is not included in the identities of the PGP key that made the signature.
unknown_key	The key that made the signature has not been registered with any user's account.
malformed_signature	There was an error parsing the signature.
invalid	The signature could not be cryptographically verified using the key whose key-id was found in the signature.
valid	None of the above errors applied, so the signature is considered to be verified.

Parameters for "Get a tag"

Headers

accept string

Setting to `application/vnd.github+json` is recommended.

Path parameters

owner string   Required

The account owner of the repository. The name is not case sensitive.

repo string   Required

The name of the repository without the `.git` extension. The name is not case sensitive.

tag_sha string   Required

HTTP response status codes for "Get a tag"

Status code	Description
`200`	OK
`404`	Resource not found

Code samples for "Get a tag"

GET `/repos/{owner}/{repo}/git/tags/{tag_sha}`

cURL   JavaScript   GitHub CLI   ⎘

```
curl -L \ -H "Accept: application/vnd.github+json" \ -H "Authorization: Bearer <YOUR-TOKEN>" \ -H "X-GitHub-Api-Version: 2022-11-28"
\ http(s)://HOSTNAME/api/v3/repos/OWNER/REPO/git/tags/TAG_SHA
```

Response

Example response   Response schema

Status: 200

```
{ "node_id": "MDM6VGFnOTQwYmQzMzYyNDhlZmFlMGY5ZWU1YmM3YjJkNWM5ODU4ODdiMTZhYw==", "tag": "v0.0.1", "sha":
"940bd336248efae0f9ee5bc7b2d5c985887b16ac", "url": "https://HOSTNAME/repos/octocat/Hello-
World/git/tags/940bd336248efae0f9ee5bc7b2d5c985887b16ac", "message": "initial version", "tagger": { "name": "Monalisa Octocat",
"email": "octocat@github.com", "date": "2014-11-07T22:01:45Z" }, "object": { "type": "commit", "sha":
"c3d0be41ecbe669545ee3e94d31ed9a4bc91ee3c", "url": "https://HOSTNAME/repos/octocat/Hello-
World/git/commits/c3d0be41ecbe669545ee3e94d31ed9a4bc91ee3c" }, "verification": { "verified": false, "reason": "unsigned",
```