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

Git commits

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

About Git commits

A Git commit is a snapshot of the hierarchy (Git tree) and the contents of the files (Git blob) in a Git repository. These endpoints allow you to read and write commit objects to your Git database on GitHub Enterprise Server.

Create a commit

Works with GitHub Apps

Creates a new Git commit object.

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 the 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 commit"

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

`message` string Required

The commit message

`tree` string Required

The SHA of the tree object this commit points to

`parents` array of strings

The SHAs of the commits that were the parents of this commit. If omitted or empty, the commit will be written as a root commit. For a single parent, an array of one SHA should be provided; for a merge commit, an array of more than one should be provided.

`author` object

Information about the author of the commit. By default, the `author` will be the authenticated user and the current date. See the `author` and `committer` object below for details.

▶ Properties of `author`

`committer` object

Information about the person who is making the commit. By default, `committer` will use the information set in `author`. See the `author` and `committer` object below for details.

▶ Properties of `committer`

`signature` string

The PGP signature of the commit. GitHub adds the signature to the `gpgsig` header of the created commit. For a commit signature to be verifiable by Git or GitHub, it must be an ASCII-armored detached PGP signature over the string commit as it would be written to the object database. To pass a `signature` parameter, you need to first manually create a valid PGP signature, which can be complicated. You may find it easier to use the command line to create signed commits.

HTTP response status codes for "Create a commit"

Status code	Description
`201`	Created
`404`	Resource not found
`422`	Validation failed, or the endpoint has been spammed.

Code samples for "Create a commit"

POST `/repos/{owner}/{repo}/git/commits`

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/comm
\n\niQIzBAABAQAdFiEESn/54jMNIrGSE6Tp6cQjvhfv7nAFAlnT71cACgkQ6cQjvhfv\n7nCWwA//XVqBKWO0zF+bZl6pggvky3Oc2j1pNFuRWZ29LXpNuD5WUGXGG209B0hI\nDkmcGk19ZKUTnEUJV2Xd0R7AW01S/YSub7OYcgBkI7qUE13FVHN5ln1KvH2al
----END PGP SIGNATURE-----\n"}'
```

Response

Example response Response schema

```
Status: 201
```

```
 { "sha": "7638417db6d59f3c431d3e1f261cc637155684cd", "node_id": "MDY6Q29tbWl0NzYzODQxN2RiNmQ1OWYzYzQzMWQzZTFmMjYxY2M2MzcxNTU2ODRjZA==", "url": "https://HOSTNAME/repos/octocat/Hello-World/git/commi
"7d1b31e74ee336d15cbd21741bc88a537ed063a0", "html_url": "https://github.com/octocat/Hello-World/commit/7d1b31e74ee336d15cbd21741bc88a537ed063a0" } ], "verification": { "verified": false, "reason":
```

Get a commit object

✔ Works with GitHub Apps

Gets a Git commit object.

To get the contents of a commit, see "Get a commit."

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 the 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 commit 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.

`commit_sha` string Required

The SHA of the commit.

HTTP response status codes for "Get a commit object"

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

Code samples for "Get a commit object"

GET /repos/{owner}/{repo}/git/commits/{commit_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/commits/COMMIT_SHA
```

Response

Example response Response schema

Status: 200

```
 { "sha": "7638417db6d59f3c431d3e1f261cc637155684cd", "node_id": "MDY6Q29tbWl0NmRjYjA5YjViNTc4NzVmMzM0ZjYxYWViZWQ2OTVlMmU0MTkzZGI1ZQ==", "url":
"https://HOSTNAME/repos/octocat/Hello-World/git/commits/7638417db6d59f3c431d3e1f261cc637155684cd", "html_url": "https://github.com/octocat/Hello-
World/commit/7638417db6d59f3c431d3e1f261cc637155684cd", "author": { "date": "2014-11-07T22:01:45Z", "name": "Monalisa Octocat", "email": "octocat@github.com" },
"committer": { "date": "2014-11-07T22:01:45Z", "name": "Monalisa Octocat", "email": "octocat@github.com" }, "message": "added readme, because im a good github citizen",
"tree": { "url": "https://HOSTNAME/repos/octocat/Hello-World/git/trees/691272480426f78a0138979dd3ce63b77f706feb", "sha": "691272480426f78a0138979dd3ce63b77f706feb" },
"parents": [ { "url": "https://HOSTNAME/repos/octocat/Hello-World/git/commits/1acc419d4d6a9ce985db7be48c6349a0475975b5", "sha":
```