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

# Issues

Use the REST API to manage issues and pull requests.

## About issues 🔗

### Custom media types 🔗

These are the supported media types for this set of endpoints:

```
application/vnd.github.raw+json
application/vnd.github.text+json
application/vnd.github.html+json
application/vnd.github.full+json
```

For more information about media types, see "Media types."

## List issues assigned to the authenticated user 🔗

List issues assigned to the authenticated user across all visible repositories including owned repositories, member repositories, and organization repositories. You can use the `filter` query parameter to fetch issues that are not necessarily assigned to you.

**Note**: GitHub's REST API considers every pull request an issue, but not every issue is a pull request. For this reason, "Issues" endpoints may return both issues and pull requests in the response. You can identify pull requests by the `pull_request` key. Be aware that the `id` of a pull request returned from "Issues" endpoints will be an *issue id*. To find out the pull request id, use the "List pull requests" endpoint.

**Parameters for "List issues assigned to the authenticated user"**

**Headers**

`accept` string

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

**Query parameters**

`filter` string

Indicates which sorts of issues to return. `assigned` means issues assigned to you. `created` means issues created by you. `mentioned` means issues mentioning you. `subscribed` means issues you're subscribed to updates for. `all` or `repos` means all issues you can see, regardless of participation or creation.

Default: `assigned`

Can be one of: `assigned`, `created`, `mentioned`, `subscribed`, `repos`, `all`

**state** string

Indicates the state of the issues to return.

Default: `open`
Can be one of: `open` , `closed` , `all`

**labels** string

A list of comma separated label names. Example: `bug,ui,@high`

**sort** string

What to sort results by.

Default: `created`
Can be one of: `created` , `updated` , `comments`

**direction** string

The direction to sort the results by.

Default: `desc`
Can be one of: `asc` , `desc`

**since** string

Only show results that were last updated after the given time. This is a timestamp in [ISO 8601](#) format: `YYYY-MM-DDTHH:MM:SSZ` .

**collab** boolean

**orgs** boolean

**owned** boolean

**pulls** boolean

**per_page** integer

The number of results per page (max 100).

Default: `30`

**page** integer

Page number of the results to fetch.

Default: `1`

## HTTP response status codes for "List issues assigned to the authenticated user"

| Status code | Description |
| --- | --- |
| 200 | OK |
| 304 | Not modified |
| 404 | Resource not found |
| 422 | Validation failed, or the endpoint has been spammed. |

## Code samples for "List issues assigned to the authenticated user"

`GET` /issues

| 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/issues
```

**Response**

| Example response | Response schema |

Status: 200

```
[ { "id": 1, "node_id": "MDU6SXNzdWUx", "url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347", "repository_url":
"https://HOSTNAME/repos/octocat/Hello-World", "labels_url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347/labels{/name}",
"comments_url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347/comments", "events_url":
"https://HOSTNAME/repos/octocat/Hello-World/issues/1347/events", "html_url": "https://github.com/octocat/Hello-World/issues/1347",
"number": 1347, "state": "open", "title": "Found a bug", "body": "I'm having a problem with this.", "user": { "login": "octocat",
"id": 1, "node_id": "MDQ6VXNlcjE=", "avatar_url": "https://github.com/images/error/octocat_happy.gif", "gravatar_id": "", "url":
```

# List organization issues assigned to the authenticated user 🔗

List issues in an organization assigned to the authenticated user.

**Note**: GitHub's REST API considers every pull request an issue, but not every issue is a pull request. For this reason, "Issues" endpoints may return both issues and pull requests in the response. You can identify pull requests by the `pull_request` key. Be aware that the `id` of a pull request returned from "Issues" endpoints will be an *issue id*. To find out the pull request id, use the "[List pull requests](#)" endpoint.

## Parameters for "List organization issues assigned to the authenticated user"

### Headers

`accept` string

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

### Path parameters

`org` string   Required

The organization name. The name is not case sensitive.

### Query parameters

`filter` string

Indicates which sorts of issues to return. `assigned` means issues assigned to you. `created` means issues created by you. `mentioned` means issues mentioning you. `subscribed` means issues you're subscribed to updates for. `all` or `repos` means all issues you can see, regardless of participation or creation.

Default: `assigned`
Can be one of: `assigned`, `created`, `mentioned`, `subscribed`, `repos`, `all`

`state` string

Indicates the state of the issues to return.

Default: `open`
Can be one of: `open` , `closed` , `all`

---

`labels`  string

A list of comma separated label names. Example: `bug,ui,@high`

---

`sort`  string

What to sort results by.

Default: `created`
Can be one of: `created` , `updated` , `comments`

---

`direction`  string

The direction to sort the results by.

Default: `desc`
Can be one of: `asc` , `desc`

---

`since`  string

Only show results that were last updated after the given time. This is a timestamp in [ISO 8601](#) format: `YYYY-MM-DDTHH:MM:SSZ` .

---

`per_page`  integer

The number of results per page (max 100).

Default: `30`

---

`page`  integer

Page number of the results to fetch.

Default: `1`

## HTTP response status codes for "List organization issues assigned to the authenticated user"

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

## Code samples for "List organization issues assigned to the authenticated user"

GET `/orgs/{org}/issues`

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/orgs/ORG/issues
```

## Response

Example response    Response schema

Status: 200

```
[ { "id": 1, "node_id": "MDU6SXNzdWUx", "url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347", "repository_url":
"https://HOSTNAME/repos/octocat/Hello-World", "labels_url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347/labels{/name}",
"comments_url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347/comments", "events_url":
"https://HOSTNAME/repos/octocat/Hello-World/issues/1347/events", "html_url": "https://github.com/octocat/Hello-World/issues/1347",
"number": 1347, "state": "open", "title": "Found a bug", "body": "I'm having a problem with this.", "user": { "login": "octocat",
"id": 1, "node_id": "MDQ6VXNlcjE=", "avatar_url": "https://github.com/images/error/octocat_happy.gif", "gravatar_id": "", "url":
```

# List repository issues 🔗

✓ Works with [GitHub Apps](#)

List issues in a repository. Only open issues will be listed.

**Note**: GitHub's REST API considers every pull request an issue, but not every issue is a pull request. For this reason, "Issues" endpoints may return both issues and pull requests in the response. You can identify pull requests by the `pull_request` key. Be aware that the `id` of a pull request returned from "Issues" endpoints will be an *issue id*. To find out the pull request id, use the "[List pull requests](#)" endpoint.

## Parameters for "List repository issues"

### 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.

### Query parameters

`milestone`  string

If an `integer` is passed, it should refer to a milestone by its `number` field. If the string `*` is passed, issues with any milestone are accepted. If the string `none` is passed, issues without milestones are returned.

`state`  string

Indicates the state of the issues to return.

Default: `open`
Can be one of: `open` , `closed` , `all`

`assignee`  string

Can be the name of a user. Pass in `none` for issues with no assigned user, and `*` for issues assigned to any user.

`creator`  string

The user that created the issue.

`mentioned`  string

A user that's mentioned in the issue.

---

**labels**  string

A list of comma separated label names. Example: `bug,ui,@high`

---

**sort**  string

What to sort results by.

Default: `created`
Can be one of: `created` , `updated` , `comments`

---

**direction**  string

The direction to sort the results by.

Default: `desc`
Can be one of: `asc` , `desc`

---

**since**  string

Only show results that were last updated after the given time. This is a timestamp in [ISO 8601](#) format: `YYYY-MM-DDTHH:MM:SSZ` .

---

**per_page**  integer

The number of results per page (max 100).

Default: `30`

---

**page**  integer

Page number of the results to fetch.

Default: `1`

## HTTP response status codes for "List repository issues"

| Status code | Description |
| --- | --- |
| 200 | OK |
| 301 | Moved permanently |
| 404 | Resource not found |
| 422 | Validation failed, or the endpoint has been spammed. |

## Code samples for "List repository issues"

GET `/repos/{owner}/{repo}/issues`

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/issues
```

## Response

Example response     Response schema

```
    Status: 200
```

```
[ { "id": 1, "node_id": "MDU6SXNzdWUx", "url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347", "repository_url":
"https://HOSTNAME/repos/octocat/Hello-World", "labels_url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347/labels{/name}",
"comments_url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347/comments", "events_url":
"https://HOSTNAME/repos/octocat/Hello-World/issues/1347/events", "html_url": "https://github.com/octocat/Hello-World/issues/1347",
"number": 1347, "state": "open", "title": "Found a bug", "body": "I'm having a problem with this.", "user": { "login": "octocat",
"id": 1, "node_id": "MDQ6VXNlcjE=", "avatar_url": "https://github.com/images/error/octocat_happy.gif", "gravatar_id": "", "url":
```

# Create an issue 🔗

✅ Works with [GitHub Apps](#)

Any user with pull access to a repository can create an issue. If [issues are disabled in the repository](#), the API returns a `410 Gone` status.

This endpoint triggers [notifications](#). Creating content too quickly using this endpoint may result in secondary rate limiting. See "[Secondary rate limits](#)" and "[Dealing with secondary rate limits](#)" for details.

## Parameters for "Create an issue"

### 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

**`title`** string or integer   Required

The title of the issue.

**`body`** string

The contents of the issue.

**`assignee`** string or null

Login for the user that this issue should be assigned to. *NOTE: Only users with push access can set the assignee for new issues. The assignee is silently dropped otherwise.* **This field is deprecated.**

**`milestone`** null or string or integer

The `number` of the milestone to associate this issue with. *NOTE: Only users with push access can set the milestone for new issues. The milestone is silently dropped otherwise.*

**`labels`** array

Labels to associate with this issue. *NOTE: Only users with push access can set labels for new issues. Labels are silently dropped otherwise.*

**assignees** array of strings

Logins for Users to assign to this issue. *NOTE: Only users with push access can set assignees for new issues. Assignees are silently dropped otherwise.*

## HTTP response status codes for "Create an issue"

| Status code | Description |
| --- | --- |
| 201 | Created |
| 400 | Bad Request |
| 403 | Forbidden |
| 404 | Resource not found |
| 410 | Gone |
| 422 | Validation failed, or the endpoint has been spammed. |
| 503 | Service unavailable |

## Code samples for "Create an issue"

POST /repos/{owner}/{repo}/issues

cURL    JavaScript    GitHub CLI

```
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/issues \ -d '{"title":"Found a bug","body":"I'\''m having a problem with
this.","assignees":["octocat"],"milestone":1,"labels":["bug"]}'
```

### Response

Example response    Response schema

Status: 201

```
{ "id": 1, "node_id": "MDU6SXNzdWUx", "url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347", "repository_url":
"https://HOSTNAME/repos/octocat/Hello-World", "labels_url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347/labels{/name}",
"comments_url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347/comments", "events_url":
"https://HOSTNAME/repos/octocat/Hello-World/issues/1347/events", "html_url": "https://github.com/octocat/Hello-World/issues/1347",
"number": 1347, "state": "open", "title": "Found a bug", "body": "I'm having a problem with this.", "user": { "login": "octocat",
"id": 1, "node_id": "MDQ6VXNlcjE=", "avatar_url": "https://github.com/images/error/octocat_happy.gif", "gravatar_id": "", "url":
```

# Get an issue 🔗

Works with [GitHub Apps](GitHub Apps)

The API returns a `301 Moved Permanently` status if the issue was [transferred](#) to another repository. If the issue was transferred to or deleted from a repository where the authenticated user lacks read access, the API returns a `404 Not Found` status. If the issue was deleted from a repository where the authenticated user has read access, the API returns a `410 Gone` status. To receive webhook events for transferred and deleted issues, subscribe to the `issues` webhook.

**Note**: GitHub's REST API considers every pull request an issue, but not every issue is a pull request. For this reason, "Issues" endpoints may return both issues and pull requests in the response. You can identify pull requests by the `pull_request` key. Be aware that the `id` of a pull request returned from "Issues" endpoints will be an *issue id*. To find out the pull request id, use the "[List pull requests](#)" endpoint.

## Parameters for "Get an issue"

### 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.

`issue_number` integer   Required

The number that identifies the issue.

## HTTP response status codes for "Get an issue"

| Status code | Description |
| --- | --- |
| `200` | OK |
| `301` | Moved permanently |
| `304` | Not modified |
| `404` | Resource not found |
| `410` | Gone |

## Code samples for "Get an issue"

`GET` /repos/{owner}/{repo}/issues/{issue_number}

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/issues/ISSUE_NUMBER
```

### Response

| Example response | Response schema |
|---|---|

```
Status: 200
```

{ "id": 1, "node_id": "MDU6SXNzdWUx", "url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347", "repository_url": "https://HOSTNAME/repos/octocat/Hello-World", "labels_url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347/labels{/name}", "comments_url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347/comments", "events_url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347/events", "html_url": "https://github.com/octocat/Hello-World/issues/1347", "number": 1347, "state": "open", "title": "Found a bug", "body": "I'm having a problem with this.", "user": { "login": "octocat", "id": 1, "node_id": "MDQ6VXNlcjE=", "avatar_url": "https://github.com/images/error/octocat_happy.gif", "gravatar_id": "", "url":

# Update an issue 🔗

✅ Works with [GitHub Apps](GitHub Apps)

Issue owners and users with push access can edit an issue.

## Parameters for "Update an issue"

### 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.

`issue_number`   integer   Required

The number that identifies the issue.

### Body parameters

`title`   null or string or integer

The title of the issue.

`body`   string or null

The contents of the issue.

`assignee`   string or null

Username to assign to this issue. **This field is deprecated.**

**state** string

The open or closed state of the issue.

Can be one of: `open` , `closed`

---

**state_reason** string or null

The reason for the state change. Ignored unless `state` is changed.

Can be one of: `completed` , `not_planned` , `reopened` , *null*

---

**milestone** null or string or integer

The `number` of the milestone to associate this issue with or use `null` to remove the current milestone. Only users with push access can set the milestone for issues. Without push access to the repository, milestone changes are silently dropped.

---

**labels** array

Labels to associate with this issue. Pass one or more labels to *replace* the set of labels on this issue. Send an empty array ( `[]` ) to clear all labels from the issue. Only users with push access can set labels for issues. Without push access to the repository, label changes are silently dropped.

---

**assignees** array of strings

Usernames to assign to this issue. Pass one or more user logins to *replace* the set of assignees on this issue. Send an empty array ( `[]` ) to clear all assignees from the issue. Only users with push access can set assignees for new issues. Without push access to the repository, assignee changes are silently dropped.

## HTTP response status codes for "Update an issue"

| Status code | Description |
| --- | --- |
| 200 | OK |
| 301 | Moved permanently |
| 403 | Forbidden |
| 404 | Resource not found |
| 410 | Gone |
| 422 | Validation failed, or the endpoint has been spammed. |
| 503 | Service unavailable |

## Code samples for "Update an issue"

**PATCH** /repos/{owner}/{repo}/issues/{issue_number}

cURL    JavaScript    GitHub CLI

```
curl -L \ -X PATCH \ -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/issues/ISSUE_NUMBER \ -d '{"title":"Found a bug","body":"I'\''m having a
problem with this.","assignees":["octocat"],"milestone":1,"state":"open","labels":["bug"]}'
```

## Response

Example response    Response schema

```
  Status: 200
```

```
{ "id": 1, "node_id": "MDU6SXNzdWUx", "url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347", "repository_url":
"https://HOSTNAME/repos/octocat/Hello-World", "labels_url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347/labels{/name}",
"comments_url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347/comments", "events_url":
"https://HOSTNAME/repos/octocat/Hello-World/issues/1347/events", "html_url": "https://github.com/octocat/Hello-World/issues/1347",
"number": 1347, "state": "open", "title": "Found a bug", "body": "I'm having a problem with this.", "user": { "login": "octocat",
"id": 1, "node_id": "MDQ6VXNlcjE=", "avatar_url": "https://github.com/images/error/octocat_happy.gif", "gravatar_id": "", "url":
```

# Lock an issue 🔗

✓ Works with [GitHub Apps](#)

Users with push access can lock an issue or pull request's conversation.

Note that, if you choose not to pass any parameters, you'll need to set `Content-Length` to zero when calling out to this endpoint. For more information, see "[HTTP verbs](#)."

## Parameters for "Lock an issue"

### 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.

**issue_number**   integer   Required

The number that identifies the issue.

### Body parameters

**lock_reason**   string

The reason for locking the issue or pull request conversation. Lock will fail if you don't use one of these reasons:
- `off-topic`
- `too heated`
- `resolved`
- `spam`

Can be one of: `off-topic`, `too heated`, `resolved`, `spam`

## HTTP response status codes for "Lock an issue"

| Status code | Description |
| --- | --- |
| 204 | No Content |

| 403 | Forbidden |
| --- | --- |
| 404 | Resource not found |
| 410 | Gone |
| 422 | Validation failed, or the endpoint has been spammed. |

## Code samples for "Lock an issue"

> **PUT** `/repos/{owner}/{repo}/issues/{issue_number}/lock`

| cURL | JavaScript | GitHub CLI | |
| --- | --- | --- | --- |

```
curl -L \ -X PUT \ -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/issues/ISSUE_NUMBER/lock \ -d '{"lock_reason":"off-topic"}'
```

### Response

```
Status: 204
```

# Unlock an issue 🔗

✓ Works with [GitHub Apps](#)

Users with push access can unlock an issue's conversation.

## Parameters for "Unlock an issue"

### 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

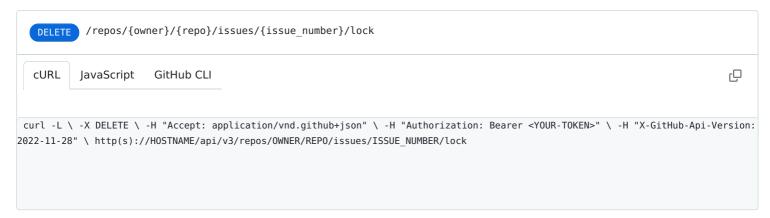The name of the repository without the `.git` extension. The name is not case sensitive.

`issue_number` integer  Required

The number that identifies the issue.

**HTTP response status codes for "Unlock an issue"**

| Status code | Description |
| --- | --- |
| `204` | No Content |
| `403` | Forbidden |
| `404` | Resource not found |

**Code samples for "Unlock an issue"**

DELETE  /repos/{owner}/{repo}/issues/{issue_number}/lock

cURL    JavaScript    GitHub CLI

```
curl -L \ -X DELETE \ -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/issues/ISSUE_NUMBER/lock
```

**Response**

```
Status: 204
```

# List user account issues assigned to the authenticated user 🔗

List issues across owned and member repositories assigned to the authenticated user.

**Note**: GitHub's REST API considers every pull request an issue, but not every issue is a pull request. For this reason, "Issues" endpoints may return both issues and pull requests in the response. You can identify pull requests by the `pull_request` key. Be aware that the `id` of a pull request returned from "Issues" endpoints will be an *issue id*. To find out the pull request id, use the "[List pull requests](#)" endpoint.

**Parameters for "List user account issues assigned to the authenticated user"**

**Headers**

`accept` string

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

**Query parameters**

`filter` string

Indicates which sorts of issues to return. `assigned` means issues assigned to you. `created` means issues created by you. `mentioned` means issues mentioning you. `subscribed` means issues you're subscribed to updates for. `all` or `repos` means all issues you can see, regardless of participation or creation.

Default: `assigned`

Can be one of: `assigned`, `created`, `mentioned`, `subscribed`, `repos`, `all`

**state** string

Indicates the state of the issues to return.

Default: `open`
Can be one of: `open` , `closed` , `all`

---

**labels** string

A list of comma separated label names. Example: `bug,ui,@high`

---

**sort** string

What to sort results by.

Default: `created`
Can be one of: `created` , `updated` , `comments`

---

**direction** string

The direction to sort the results by.

Default: `desc`
Can be one of: `asc` , `desc`

---

**since** string

Only show results that were last updated after the given time. This is a timestamp in [ISO 8601](#) format: `YYYY-MM-DDTHH:MM:SSZ` .

---

**per_page** integer

The number of results per page (max 100).

Default: `30`

---

**page** integer

Page number of the results to fetch.

Default: `1`

## HTTP response status codes for "List user account issues assigned to the authenticated user"

| Status code | Description |
| --- | --- |
| `200` | OK |
| `304` | Not modified |
| `404` | Resource not found |

## Code samples for "List user account issues assigned to the authenticated user"

> **GET** `/user/issues`

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/user/issues
```

**Response**

**Example response** | Response schema

```
Status: 200
```

```
[ { "id": 1, "node_id": "MDU6SXNzdWUx", "url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347", "repository_url":
"https://HOSTNAME/repos/octocat/Hello-World", "labels_url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347/labels{/name}",
"comments_url": "https://HOSTNAME/repos/octocat/Hello-World/issues/1347/comments", "events_url":
"https://HOSTNAME/repos/octocat/Hello-World/issues/1347/events", "html_url": "https://github.com/octocat/Hello-World/issues/1347",
"number": 1347, "state": "open", "title": "Found a bug", "body": "I'm having a problem with this.", "user": { "login": "octocat",
"id": 1, "node_id": "MDQ6VXNlcjE=", "avatar_url": "https://github.com/images/error/octocat_happy.gif", "gravatar_id": "", "url":
```

**Legal**