



Filter by keyword



# requestGraph

## ON THIS PAGE

### Method signature

[Experimental endpoints](#)[Parameters](#)[Authenticated example](#)

You can use `requestGraph` to make a request to the [Atlassian platform GraphQL API](#), from within your Forge app. This is suitable for Atlassian apps that expose GraphQL API endpoints, such as Compass.



Alternatively, Compass Forge apps can use [GraphQL API Toolkit](#) to execute pre-written queries (instead of using `requestGraph`).

The client includes an `Authorization` header based on the context-bound fetch call.

To use `requestGraph`, import the Forge API package in your app:

```
1 import api, { route } from '@forge/api';
```

To check the status of a request, use `status` as shown below:

```
1 const result = await api
2   .asApp()
3   .requestGraph(
4     query, variables, headers
5   );
6 const status = result.status;
```

## Method signature



When making an authenticated API call, you can specify whether to use the app ( `asApp` ) or its user ( `asUser` ) as the context:

Method	Description
<code>api.asApp()</code>	<p>Use this method to call an Atlassian API as the app, using the app user. If your app has all the required permissions, the request will work regardless of who uses the app.</p> <p>When using <code>asApp</code> , you can use the <a href="#">Authorize API</a> to verify user permissions before making the request.</p>
<code>api.asUser()</code>	<p>Use this method to call an Atlassian API as the user of the app.</p> <p>If the user initiating the request has not granted permission to the app, the request is made without a valid <code>Authorization</code> header.</p> <p>If the API returns a <code>401</code> and there isn't a valid <code>Authorization</code> header, then an <code>Error</code> is thrown.</p> <ul style="list-style-type: none"><li>• If this error isn't caught, the user is prompted to grant access to the app.</li><li>• If the user grants access, the app is re-initialized.</li></ul>

## Experimental endpoints

In the [GraphQL API reference](#), some endpoints may still be experimental. The queries and mutations for these fields are documented with a **Field Lifecycle** section that includes the following notice:

```
1 This field is in the 'EXPERIMENTAL' lifecycle stage
2
```

The **Field Lifecycle** section will include specific instructions on how to query the field. In particular, any requests to experimental Compass endpoints need to include the `X-ExperimentalApi: compass-beta` header.

## Parameters

Name	Type	Description
<code>query</code>	<code>string</code>	The GraphQL query or mutation.
<code>variables</code>	<code>object</code>	Variables to be used in the GraphQL query.



## Authenticated example

Fetch the name of the current user:

```
1  const query = "query { me { user { name } } }";
2  const variables = { "key1": "value1", "key2": "value2" };
   const headers = { "key1": "value1", "key2": "value2" };

   api.[asApp | asUser]().requestGraph(query, variables, headers);
```

Successful queries return their result in the `data` object. Any errors are returned as an array in the `errors` object.

Rate this page: ☆ ☆ ☆ ☆ ☆

[Changelog](#)[System status](#)[Privacy](#)[Developer Terms](#)[Trademark](#)

© 2026 Atlassian