Group

Path : `/{group-id}`

Represents a Workplace group. This `/{group-id}` node returns a single group.

Reading

You can read information about a group by making a Graph API GET request to `/{group-id}`.

Permissions

Reading Group node requires either **Read content** or **Read company visible content** permission for closed/secret groups and open groups respectively.

Fields

Field Name	Description	Data Type
id	The group ID	string
cover	Information about the Group's cover photo.	CoverPhoto
description	A brief description of the group.	string
icon	The URL for the group's icon.	url
is_workplace_default	Indicates whether the group is a default Workplace group (read only).	url
name	The name of the group.	string
owner	The member that created this group.	User
privacy	This privacy setting of this group. Possible values: • CLOSED • OPEN • SECRET	string
updated_time	The last time the group was updated (this includes changes in the group's properties and changes in posts and comments if the session user can see them).	datetime
archive	Indicates whether the group was archived (read only).	boolean
is_community	Indicates whether the group is a community and can therefore contain other groups (read only).	boolean

Publishing

You can't publish using this edge. In order to create a group, publish to the /community/groups edge.

Deleting

You can't delete a group using this node. Removing the last member of a group will delete that group automatically.

Updating

You can update a group by making a Graph API POST request to `/{group-id}` and passing values for the fields to be updated.

Permissions

Making updates to a group node requires the **Manage groups** and **Manage content** permissions.

Fields

Field Name	Description	Type
name	The name of the group being updated	string
cover_url	A URL containing an image for the group cover photo.	string
privacy	The privacy setting of the group. Possible values: • CLOSED • OPEN • SECRET	string

Edges

Edge Name	Description
/admins	The admins of a Workplace group.
/albums	The photo albums in a Workplace group.
/docs	The docs in a Workplace group.
/events	The events in a Workplace group.
/feed	The posts in a Workplace group, arranged like a feed.
/files	The files shared into a Workplace group.
/groups	The groups contained within this community (if this group is a community)
/member_requests	The pending membership requests for groups that have membership approvals enabled.
/members	The members of a Workplace group. This edge exposes: • administrator: Shown if this person is an admin of the group • joined: Shown when this user joined the group • moderator: Shown if this person is a moderator of the group • added_by: Shown who added this user to the group
/moderators	The moderators of a Workplace group.

Examples

Get group id, name, and privacy:

```
HTTP
GET graph.facebook.com
  /{group-id}?fields=id,name,privacy
```

Get group members with name, id, and join date:

```
HTTP
GET graph.facebook.com
  /{group-id}/members?fields=name,id,joined
```

Get a group's admins and moderators in a single call:

```
HTTP
GET graph.facebook.com
  /{group-id}?fields=admins,moderators
```

Get group docs:

```
HTTP
GET graph.facebook.com
  /{group-id}/docs
```

Get group posts:

```
HTTP
GET graph.facebook.com
  /{group-id}/feed
```

Get group posts including additional attachments like videos, images, files, or polls:

```
HTTP
GET graph.facebook.com
  /{group-id}/feed?fields=attachments
```

ⓘ Poll options are listed in descending order according to vote count for each option.

Get a list of group members, along with their join date:

```
HTTP
GET graph.facebook.com
  /{group-id}/members?fields=name,joined
```

Add a member to a group by id:

```
HTTP
POST graph.facebook.com
  /{group-id}/members/{member-id}
```

Add a member to a group by email:

```
HTTP
POST graph.facebook.com
  /{group-id}/members?email=michael%40example.com
```

ⓘ When including email addresses in the URL for a request, ensure that the email addresses is URL encoded. Example: michael@example.com becomes michael%40example.com.

⚠ Removing the last member from a group will schedule that group for deletion.

Remove a member from a group by id:

```
HTTP
DELETE graph.facebook.com
  /{group-id}/members/{member-id}
```

Remove a member from a group by email:

```
HTTP
DELETE graph.facebook.com
  /{group-id}/members?email=michael%40example.com
```

ⓘ When including email addresses in the URL for a request, ensure that the email addresses is URL encoded. Example: michael@example.com becomes michael%40example.com.

Promote a member to admin of a group:

```
HTTP
POST graph.facebook.com
  /{group-id}/admins/{user-id}
```

Demote an admin to member of a group:

```
HTTP
DELETE graph.facebook.com
  /{group-id}/admins/{user-id}
```

Create a new Event in a group:

```
HTTP
POST graph.facebook.com
  /{group-id}/events
    ?name=NewEvent
    &start_time=2017-03-02T17:00:04+00:00
    &end_time=2017-03-02T19:00:04+00:00
    &description=TestEvent+description
    &location=Boardroom
```

Upload a new photo (via binary) to a group:

```
HTTP
```

```
POST graph.facebook.com
  /{group-id}/photos?source={image-data}
```

Upload a new photo (via url) to a group:

`HTTP`

```
POST graph.facebook.com
  /{group-id}/photos?url={image-data}
```

Create a new group doc with title and body:

`HTTP`

```
POST graph.facebook.com
  /{group-id}/docs?title={title}&body={body}
```

Get a list of groups within this community:

`HTTP`

```
POST graph.facebook.com
  /{group-id}/groups
```