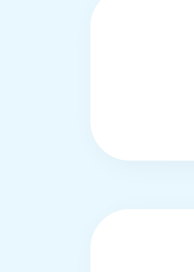
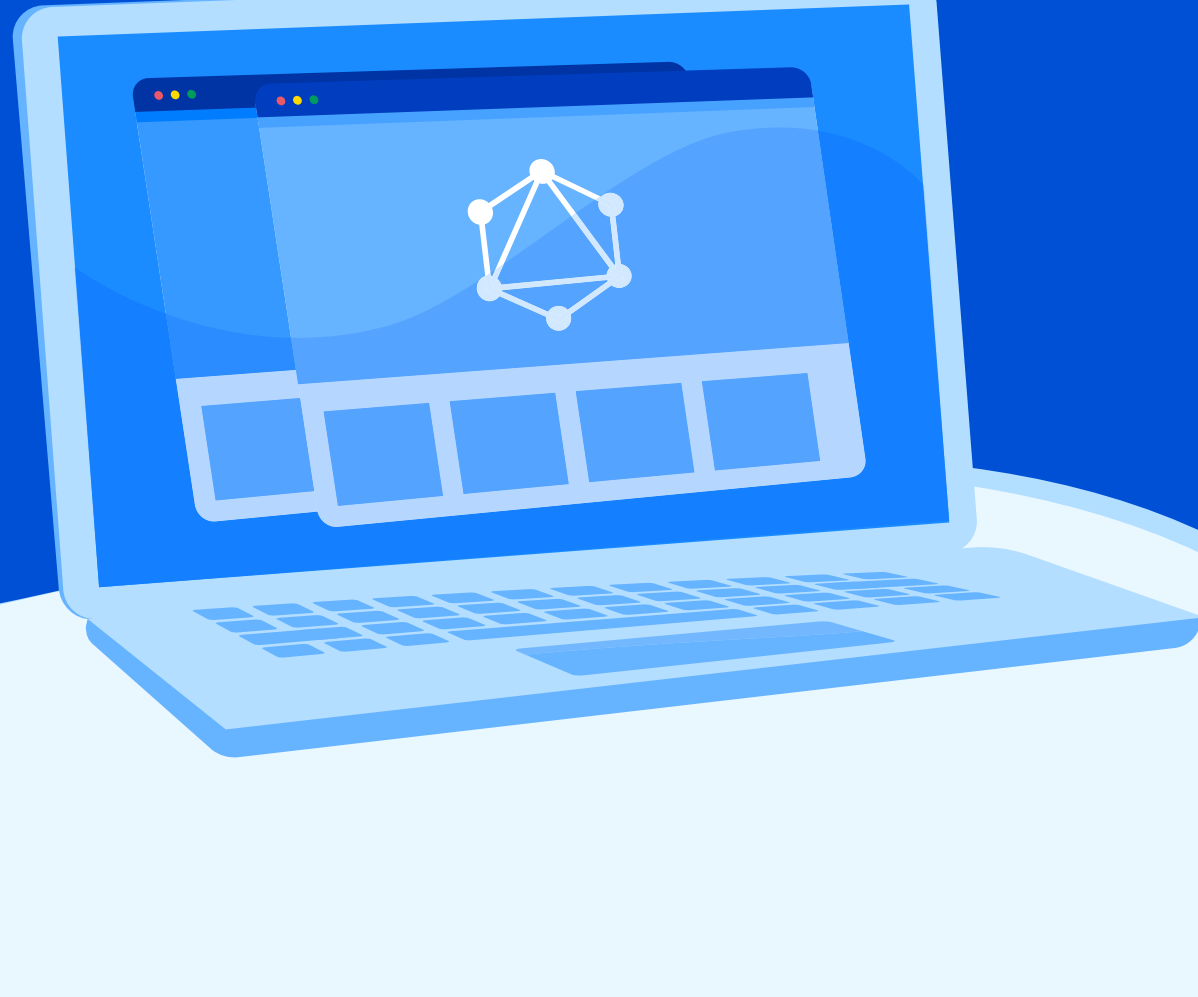


# GraphQL API cheat sheet

GraphQL is a query language that gives developers flexibility in retrieving data from APIs. This cheat sheet highlights the most commonly used GraphQL features and tools.



## GraphQL tools and access

<b>BASIC API</b>	→	Read-only API for content from Contentful.
<b>BASE URL</b>	→	<a href="https://graphql.contentful.com">https://graphql.contentful.com</a>
<b>CONTENT API</b>	→	<a href="https://graphql.contentful.com/content/v1/spaces/[SPACE_ID]">https://graphql.contentful.com/content/v1/spaces/[SPACE_ID]</a>
<b>GRAPHQL</b> An in-browser GraphQL IDE.	→	<a href="https://graphql.contentful.com/content/v1/spaces/[SPACE_ID]/explore?access_token=[CDA_TOKEN]">https://graphql.contentful.com/content/v1/spaces/[SPACE_ID]/explore?access_token=[CDA_TOKEN]</a>
<b>GRAPHQL PLAYGROUND</b> A Contentful App that allows you to run queries within Contentful's web interface.	→	<a href="https://www.contentful.com/marketplace/app/graphql-playground/">https://www.contentful.com/marketplace/app/graphql-playground/</a>



**Tip:** You can access your space ID and [Content Delivery API](#) token in your Contentful space settings.



## Contentful GraphQL queries

### Single entry query

Retrieve a single entry.

```
query {
  blogPost (id: "1234") {
    title
  }
}
```

### Collection query

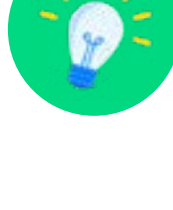
Retrieve multiple entries.

```
query {
  blogPostCollection {
    items {
      title
    }
  }
}
```

### Preview query

Include draft entries.

```
query {
  blogPostCollection (preview: true) {
    items {
      title
    }
  }
}
```



**Tip:** You'll need to use your [Content Preview API](#) token to access draft content.

### Reference queries

#### SINGLE REFERENCE TYPES

Query a reference field that only accepts a specified entry type.

```
query {
  blogPostCollection {
    items {
      author {
        name
      }
    }
  }
}
```

#### Various reference types

Query a reference field that accepts multiple entry types.

```
query {
  blogPostCollection (limit: 10) {
    items {
      author {
        ... on Person {
          name
        }
        ... on Company {
          name
        }
      }
    }
  }
}
```



**Tip:** Queries with many references could hit [query complexity limits](#). You can add limits to your queries to lower the complexity cost.



## GraphQL basics

### Reference queries

Reuse queries by passing down variables.

#### DEFINE VARIABLE

```
{ "author": "Contentful" }
```

#### USE VARIABLE IN A QUERY

```
query ($author: String!) {
  authorCollection(where: {name: $author}) {
    items {
      name
    }
  }
}
```

### Named queries

Provide unique query names to organize multiple queries.

```
query blogPosts {
  blogPostCollection {
    items {
      title
    }
  }
}

query sponsorships {
  sponsorCollection {
    items {
      name
    }
  }
}
```

### Aliases

Provide alternative names, or aliases, for query results to avoid conflicts when fetching from the same field or collection.

```
query {
  allBlogPosts: blogPostCollection {
    items {
      title
    }
  }
  sponsoredBlogPosts: blogPostCollection (where: {sponsored_exists: true}) {
    items {
      title
    }
  }
}
```

### Fragments

Reusable field groups can be shared between multiple fields and queries.

```
query {
  blogPostCollection {
    items {
      ...blogFields
    }
  }
}

fragment blogFields on BlogPost {
  title
  date
  author {
    name
    githubUsername
  }
}
```

### Directives

Include or skip a field in a query based on a variable value.

#### CREATE VARIABLE

```
{ "isApproved": false }
```

#### USE VARIABLE WITH DIRECTIVES

```
query ($isApproved: Boolean!) {
  blogPostCollection {
    items {
      title
      sponsorMessage @include(if: $isApproved)
    }
  }
}
```

### Request

Example of how to request data using GraphQL

```
const query = `
  blogPostCollection {
    items {
      title
    }
  }
`

const response = await fetch(
  `https://graphql.contentful.com/content/v1/spaces/${SPACE_ID}`,
  {
    method: "POST",
    headers: {
      Authorization: `Bearer ${ACCESS_TOKEN}`,
      content-type: "application/json",
    },
    body: JSON.stringify({ query }),
  },
).then((response) => response.json());
```

### Variables

Create dynamic queries and add type safety.

#### DEFINE VARIABLES

```
const variables = {author: "Contentful" }
```

#### USE VARIABLE IN A QUERY

```
const query = `query ($author: String!) {
  blogPostCollection(where: {name: $author}) {
    items {
      title
    }
  }
}`
```

#### USE VARIABLE IN API REQUEST

```
const response = await fetch(
  `https://graphql.contentful.com/content/v1/spaces/${SPACE_ID}`,
  {
    method: "POST",
    headers: {
      Authorization: `Bearer ${ACCESS_TOKEN}`,
      content-type: "application/json",
    },
    body: JSON.stringify({ query, variables }),
  },
).then((response) => response.json());

console.log(response);
```



**Tip:** Check out our blog for a detailed walkthrough of how to [utilize variables in GraphQL](#) in requests.



## Additional resources

- [Explore GraphQL API Documentation](#)
- [Watch our GraphQL video series](#)
- [Introduction to GraphQL](#)

Have additional questions about using GraphQL with Contentful?

Join the [Contentful Slack Community](#) for assistance, tips and tricks