


GraphQL Read all Books


GraphiQL  Prettify Merge Copy History

```
1 query {
2   books {
3     id
4     name
5     author {
6       id
7       name
8     }
9   }
10 }
11
```

```
{
  "data": {
    "books": [
      {
        "id": 1,
        "name": "Harry Potter and the Chamber of Secrets",
        "author": {
          "id": 1,
          "name": "J. K. Rowling"
        }
      },
      {
        "id": 2,
        "name": "Harry Potter and the Prisoner of Azkaban",
        "author": {
          "id": 1,
          "name": "J. K. Rowling"
        }
      },
      {
        "id": 3,
        "name": "Harry Potter and the Goblet of Fire",
        "author": {
          "id": 1,
          "name": "J. K. Rowling"
        }
      },
      {
        "id": 4,
        "name": "The Merchant of Venice",
        "author": {
          "id": 2,
          "name": "Shakespeare"
        }
      },
      {
        "id": 5,
        "name": "Antony and Cleopatra",
        "author": {
          "id": 2,
          "name": "Shakespeare"
        }
      },
      {
        "id": 6,
        "name": "Geetanjali",
        "author": {
          "id": 3,
          "name": "Rabindra Nath Tagore"
        }
      }
    ]
  }
}
```

QUERY VARIABLES


GraphQL Read Books by id

GraphiQL  Prettify Merge Copy History

```
1 query {
2   book(id: 6) {
3     id
4     name
5     author {
6       id
7       name
8     }
9   }
10 }
11
```

```
{
  "data": {
    "book": {
      "id": 6,
      "name": "Geetanjali",
      "author": {
        "id": 3,
        "name": "Rabindra Nath Tagore"
      }
    }
  }
}
```


GraphQL create books

GraphiQL  Prettify Merge Copy History

```
1 mutation {
2   addBook(name: "New added Book", authorId: 3) {
3     id
4     name
5     author {
6       id
7       name
8     }
9   }
10 }
11
```

```
{
  "data": {
    "addBook": {
      "id": 7,
      "name": "New added Book",
      "author": {
        "id": 3,
        "name": "Rabindra Nath Tagore"
      }
    }
  }
}
```


GraphQL update books

GraphiQL  Prettify Merge Copy History

```
1 mutation {
2   updateBookName(id: 7, newName: "Updated new added book") {
3     id
4     name
5   }
6 }
7
```

```
{
  "data": {
    "updateBookName": {
      "id": 7,
      "name": "Updated new added book"
    }
  }
}
```

GraphQL delete books

GraphiQL  Prettify Merge Copy History

```
1 mutation {
2   deleteBook(id: 7)
3 }
4
```

```
{
  "data": {
    "deleteBook": "Book with Id 7 deleted successfully"
  }
}
```

GraphQL Read all Authors

The screenshot shows the GraphQL IDE interface. On the left, a query is written in a monospace font with line numbers 1 through 11. The query is:

```
1 query {  
2   authors {  
3     id  
4     name  
5     books {  
6       id  
7       name  
8     }  
9   }  
10 }  
11
```

 A dropdown menu is open below the query, showing options: `book`, `books`, `authors`, `author`, `__typename`, `__schema`, `__type`, and `Book A Single Book`. The `book` option is highlighted. On the right, the JSON response is displayed, showing a list of authors and their books. The response is:

```
{  
  "data": {  
    "authors": [  
      {  
        "id": 1,  
        "name": "J. K. Rowling",  
        "books": [  
          {  
            "id": 1,  
            "name": "Harry Potter and the Chamber of Secrets"  
          },  
          {  
            "id": 2,  
            "name": "Harry Potter and the Prisoner of Azkaban"  
          },  
          {  
            "id": 3,  
            "name": "Harry Potter and the Goblet of Fire"  
          }  
        ]  
      },  
      {  
        "id": 2,  
        "name": "Shakespeare",  
        "books": [  
          {  
            "id": 4,  
            "name": "The Merchant of Venice"  
          },  
          {  
            "id": 5,  
            "name": "Antony and Cleopatra"  
          }  
        ]  
      },  
      {  
        "id": 3,  
        "name": "Rabindra Nath Tagore",  
        "books": [  
          {  
            "id": 6,  
            "name": "Geetanjali"  
          }  
        ]  
      }  
    ]  
  }  
}
```

GraphQL Read Authors by id

The screenshot shows the GraphQL IDE interface. The left pane contains a query:

```
1 query {  
2   author(id: 2) {  
3     name  
4   }  
5 }  
6
```

. The right pane shows the JSON response:

```
{  
  "data": {  
    "author": {  
      "name": "Shakespeare"  
    }  
  }  
}
```

. The top bar includes a play button and buttons for 'Prettify', 'Merge', 'Copy', and 'History'.

GraphQL create Authors

The screenshot shows the GraphQL IDE interface. The left pane contains a mutation:

```
1 mutation {  
2   addAuthor(name: "New added Author") {  
3     id  
4     name  
5   }  
6 }  
7
```

. A tooltip is visible over line 4, showing 'Mutation.addAuthor: Author' and 'Add an author'. Below the mutation, a blue button labeled 'query' is visible. The right pane shows the JSON response:

```
{  
  "data": {  
    "addAuthor": {  
      "id": 4,  
      "name": "New added Author"  
    }  
  }  
}
```

. The top bar includes a play button and buttons for 'Prettify', 'Merge', 'Copy', and 'History'.

GraphQL update Authors

The screenshot shows the GraphQL IDE interface. The left pane contains a mutation:

```
1 mutation {  
2   updateAuthorName(id: 4, newName: "Updated new added Author Name") {  
3     id  
4     name  
5   }  
6 }  
7
```

. The right pane shows the JSON response:

```
{  
  "data": {  
    "updateAuthorName": {  
      "id": 4,  
      "name": "Updated new added Author Name"  
    }  
  }  
}
```

. The top bar includes a play button and buttons for 'Prettify', 'Merge', 'Copy', and 'History'.

GraphQL delete Authors

The screenshot shows the GraphQL IDE interface. The left pane contains a mutation:

```
1 mutation {  
2   deleteAuthor(id: 4)  
3 }  
4
```

. The right pane shows the JSON response:

```
{  
  "data": {  
    "deleteAuthor": "Author with ID 4 deleted successfully"  
  }  
}
```

. The top bar includes a play button and buttons for 'Prettify', 'Merge', 'Copy', and 'History'.

Rest read all books

GET

/books Returns all the books

Parameters

No parameters

Execute

Clear

Responses

Curl

Request URL

Server response

Code

Details

200

Response body

Download

```
curl -X 'GET' \
  'http://localhost:4000/books' \
  -H 'accept: application/json'
```

```
http://localhost:4000/books
```

```
200
{
  "name": "Harry Potter and the Chamber of Secrets",
  "authorId": 1
},
{
  "id": 2,
  "name": "Harry Potter and the Prisoner of Azkaban",
  "authorId": 1
},
{
  "id": 3,
  "name": "Harry Potter and the Goblet of Fire",
  "authorId": 1
},
{
  "id": 4,
  "name": "The Merchant of Venice",
  "authorId": 2
},
{
  "id": 5,
  "name": "Antony and Cleopatra",
  "authorId": 2
},
{
  "id": 6,
  "name": "Geetanjali",
  "authorId": 3
}
}
```

Rest read books by id

GET

/books/{id} Returns a single book by ID

Parameters

Cancel

Name

Description

id * required

integer

(path)

2

Execute

Clear

Responses

Curl

Request URL

Server response

Code

Details

200

Response body

Download

```
curl -X 'GET' \
  'http://localhost:4000/books/2' \
  -H 'accept: application/json'
```

```
http://localhost:4000/books/2
```

```
200
{
  "id": 2,
  "name": "Harry Potter and the Prisoner of Azkaban",
  "authorId": 1
}
```

Rest create books



Rest update books

PUT

/books/{id} Update a book

Cancel

Reset

Parameters

Name	Description
id * required integer (path)	ID of the book to update

7

Request body * required

application/json

```
{
  "title": "updated new book",
  "author": "3"
}
```

Execute

Clear

Responses

Curl

```
curl -X 'PUT' \
  'http://localhost:4000/books/7' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "title": "updated new book",
    "author": "3"
  }'
```

Request URL

http://localhost:4000/books/7

Server response

Code	Details
200	<div>Response body</div> <pre>{ "id": 7, "title": "updated new book", "author": "3" }</pre> <div>Download</div>

Rest delete books

DELETE

/books/{id} Delete a book

Parameters

Cancel

Name	Description
id * required integer (path)	ID of the book to delete

Execute

Clear

Responses

Curl

```
curl -X 'DELETE' \
'http://localhost:4000/books/7' \
-H 'accept: */*'
```

Request URL

```
http://localhost:4000/books/7
```

Server response

Code	Details
200	<div>Response body<div>Book deleted successfully</div><div>CopyDownload</div></div>