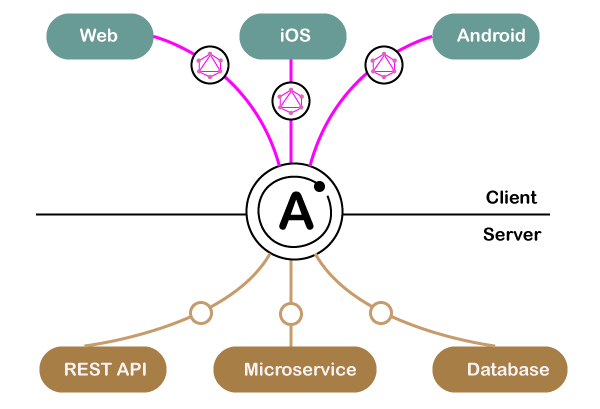
Apollo Server Installation

A GraphQL server is used to accept incoming requests on an endpoint, interpret the request, and retrieve the data necessary to fulfill the client's needs.

There are a lot of different GraphQL server implementations for many programming languages. Apollo server is one of them.

What is Apollo Server?

Apollo Server is a spec-compliant, open-source GraphQL server implementation for [JavaScript](https://www.javatpoint.com/javascript-tutorial) (particularly for the Node.js platform). It is compatible with any [GraphQL](https://www.javatpoint.com/graphql) client, including Apollo Client. Apollo Server provides the best suitable environment to build a production-ready, self-documenting GraphQL API which can be used to retrieve data from any source.



Apollo server supports the following popular [Node.js frameworks](https://www.javatpoint.com/nodejs-tutorial):

* Express
* Hapi
* Koa
* Restify etc.

Usage of Apollo Server

* Apollo server provides a way to describe our data using a schema.
* Apollo server provides the framework for resolvers (Resolvers are the functions write to fetch the data to fulfill the request of the client).
* Apollo server facilitates handling authentication for our API.

Getting Started with Apollo Server

Let's install and configure Apollo Server for the GraphQL project. Here, we shall learn:

* Basic understanding of GraphQL principles
* How to Define a GraphQL schema that represents the structure of your data set
* Run Apollo Server that facilitates you to execute queries against your schema.

Prerequisite

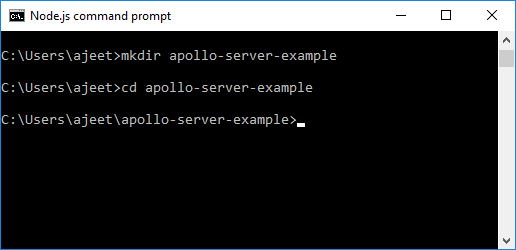
* Before going further, you must be familiar with the command line and JavaScript.
* Must have installed the recent version of Node.js (8+)

Follow the steps given below

**Step1: Create a new project**

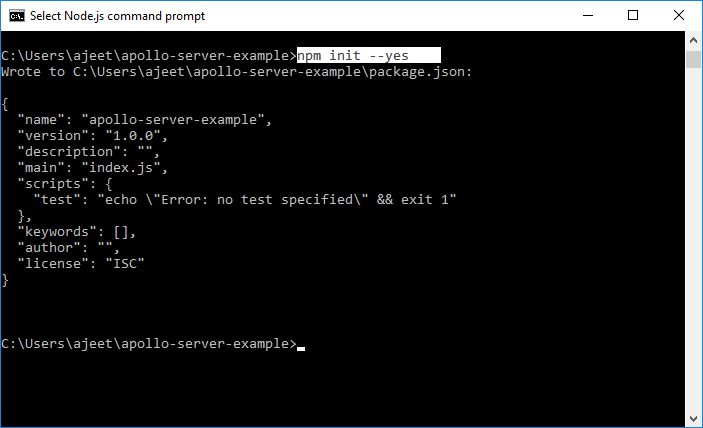
Open the Node.js command prompt and create a directory for a new project "apollo-server-example" and navigate to the folder by using the command prompt.

1. mkdir apollo-server-example
2. cd apollo-server-example



Create a package.json file within your project directory by using the following command:

1. npm init --yes



Now your project directory contains the package.json file.

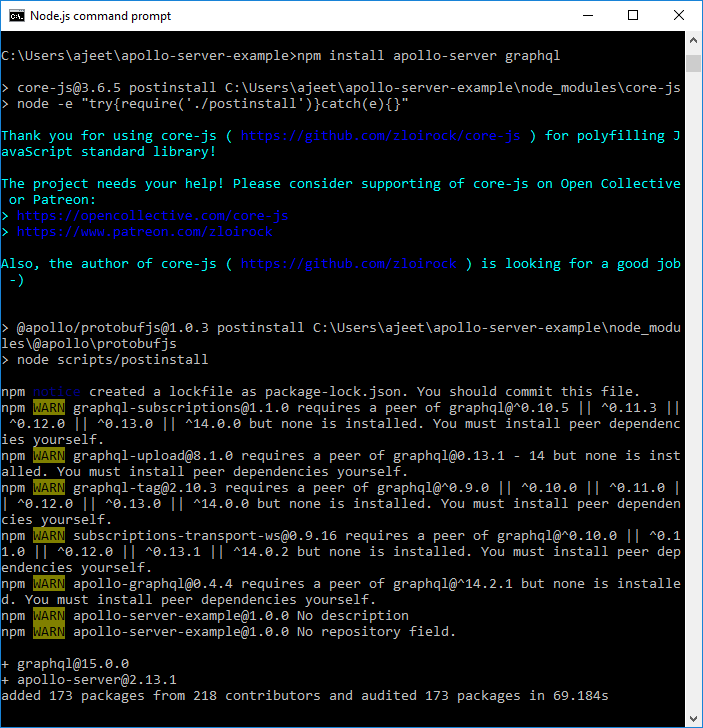
**Step 2: Install dependencies**

The Apollo server's applications require two top-level dependencies:

* apollo-server: It is the core library for Apollo Server, which facilitates us to define the shape of our data and tell how to fetch it.
* graphql: This library is used to build a GraphQL schema and execute queries against it.

Run the following command to install both the above dependencies within your project folder.

1. npm install apollo-server graphql



Now, create an empty file named **server.js** in your project's root directory:

**Step 3: Define GraphQL Schema**

We need to define a [schema](https://www.javatpoint.com/graphql-schema) in Every GraphQL server, including Apollo Server. The schema defines the structure of data queried by the client.

Here, we'll create a server for querying a list of companies by their name and owner.

Open **server.js** file in an editor and paste the following code:

1. **const** { ApolloServer, gql } = require('apollo-server');
2. **const** typeDefs = gql`
3. type Company {
4. name: String
5. owner: String
6. }
7. type Query {
8. companies: [Company]
9. }
10. `;

**Step 4: Define your data set**

Above, we have defined the structure of our data. Now, we have to define the data itself. Apollo Server can fetch data from any of your connected sources such as a database, a REST API, a static object storage service, or even another GraphQL server.

Paste the following code to the bottom of server.js:

1. **const** companies = [
2. {
3. name: 'Infosys',
4. owner: 'N.R Narayan Murthy',
5. },
6. {
7. name: 'Wipro',
8. owner: 'Mohamed Hasham Premji',
9. },
10. {
11. name: 'Reliance Industries',
12. owner: 'Dhirubhai Ambani',
13. },
14. {
15. name: 'Bajaj Auto',
16. owner: 'Jamnalal Bajaj',
17. },
18. ];

**Step 5: Define a resolver**

We have defined our data set. Now, we have to create a resolver to make Apollo server know that use this data set while executing the query.

Resolvers are used to tell Apollo Server how to fetch the data associated with a particular type.

Open the of server.js file and paste the following code to the bottom:

1. **const** resolvers = {
2. Query: {
3. companies: () => companies,
4. },
5. };

**Step 6: Create an instance of ApolloServer**

We have defined our schema, data set, and resolver above. Now, we have to provide this information to Apollo Server when we initialize it.

Now, paste the following code to the bottom of **server.js:**

1. **const** server = **new** ApolloServer({ typeDefs, resolvers });
2. server.listen().then(({ url }) => {
3. console.log(`Server ready at ${url}`);
4. });

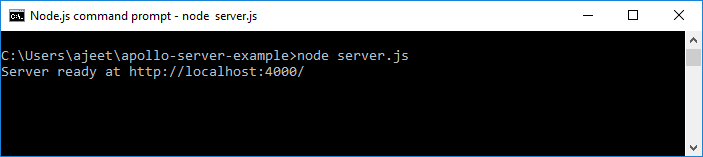
**Now, the complete server.js file will look like this:**

1. **const** { ApolloServer, gql } = require('apollo-server');
2. **const** typeDefs = gql`
3. type Company {
4. name: String
5. owner: String
6. }
7. type Query {
8. companies: [Company]
9. }
10. `;
11. **const** companies = [
12. {
13. name: 'Infosys',
14. owner: 'N.R Narayan Murthy',
15. },
16. {
17. name: 'Wipro',
18. owner: 'Mohamed Hasham Premji',
19. },
20. {
21. name: 'Reliance Industries',
22. owner: 'Dhirubhai Ambani',
23. },
24. {
25. name: 'Bajaj Auto',
26. owner: 'Jamnalal Bajaj',
27. },
28. ];
29. **const** resolvers = {
30. Query: {
31. companies: () => companies,
32. },
33. };
34. **const** server = **new** ApolloServer({ typeDefs, resolvers });
35. server.listen().then(({ url }) => {
36. console.log(`Server ready at ${url}`);
37. });

**Step 7: Start the Apollo server**

Our server is ready to start. Now, run the following command to start the server:

1. node server.js



You can see that the server is started.

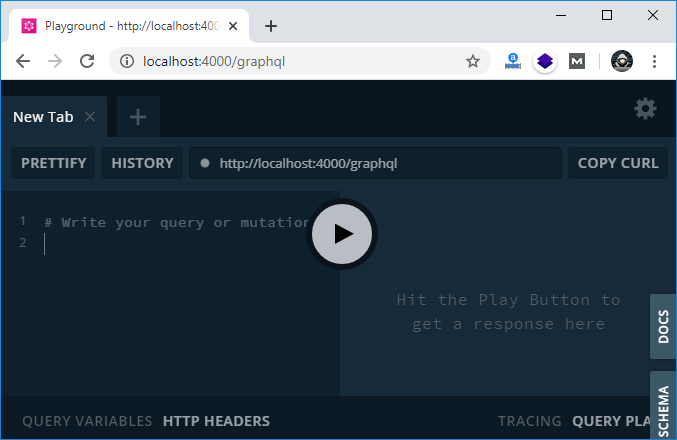
**Step 8: Execute Our First Query**

We are ready to execute our first GraphQL query on the Apollo server. Here, we shall use a tool called GraphQL Playground.

Go to the following localhost url on your browser to run GraphQL Playground

http://localhost:4000

(Apollo Server hosts GraphQL Playground automatically on the local host when the server is started.)



Use the following GraphQL query string to execute the companies query:

1. {
2. companies {
3. name
4. owner
5. }
6. }

**Output:**

{

"data": {

"companies": [

{

"name": "Infosys",

"owner": "N.R Narayan Murthy"

},

{

"name": "Wipro",

"owner": "Mohamed Hasham Premji"

},

{

"name": "Reliance Industries",

"owner": "Dhirubhai Ambani"

},

{

"name": "Bajaj Auto",

"owner": "Jamnalal Bajaj"

}

]

}

}

