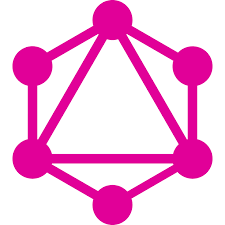
**Graph Query Language**.

* GraphQL is a query language for APIs and a runtime for fulfilling those queries with your existing data
* A GraphQL query is a string that is sent to a server to be interpreted and fulfilled, which then returns JSON back to the client.
* Client decides what data to receive not the server like in REST
* Scehma based
* Developed by Facebook in 2012
* Open sourced in 2005

**Traditional Method**

**REST Endpoints**

/posts

/postInfo

/postJustTitle

/postsByAuthor

/postByTag

/Commenstonpost

Client

DataBase

**Data Retrive**

End-Points

**Front End Application [ReactJS]**

**Back End Application [NodeJS]**

**GraphQL Method**

GraphQL API

{---------

-----}

Client

DataBase

Data Retrive

End-Points

**Back End Application [NodeJS]**

**Front End Application [ReactJS]**

**GraphQL Operations**

**Subcriptions**

Receive Data in Real-time

Subscription {

onCreateComment { id }

}

**Mutations**

Write Data

Mutation {

createEvent (name: “venkat”) { Id

}

}

**Queries**

Read Data

Query {

getEvent(id: 1) {

id

name}

}

**How to use GraphQL?**

**Schema**

(Define your data)

Type Query {

listEvents: [Event]

}

Type Event {

Id: ID!

Name: string!

When: string!

Where: string! }

}

**Get Exactly what you asked for**

{

“id”: “1”

“name”: “Yoga”

“when”: “tomorrow”

}

{

“id”: “2”

“name”: Gym”

“when”: “tomorrow”

}

**Invoke operation**

Query {

listsEvents {

id

name

when

}

}

**Architecture**

Query parser Parser

Schema

Resolverr function

Apollo server

DB

Nodejs | Express

REACTJS | Javascript

GrapghiQL or Grapgh playground

Apollo Client

Query

Json data

**Back End Application [NodeJS**]

**Front End Application [ReactJS]**

**Queries :** Read data

**Mutation** : Create, update & delete the data

**Subscription**: Real time data fetch

**Schema**: Defines server API

**Resolver**: Executes Schema

**Apolo**: Platform for GraphQL API

Companies using Apolo : **,**

* **Airbnb**
* **Twitch T**
* **The New York Times**
* **KLM, Medium**

**Who Using GrapghQL ?**

**STACKSHARE**

**STARTBUCKS**

**FACEBOOK**

**TWITTER**

**PAYPAL**

**GIT-HUB**

**SHOPIFY**

**COURSERA**

**THE NEW YORK TIMES**

**MICROSOFT**

**YELP**

**CIRCLECI**

**Available Tools in the market**

|  |  |  |
| --- | --- | --- |
| **GrapghQL Servers** | **GrapghQL Client** | **GrapghQL IDE** |
|  |  |  |
| Apolo server | Apolo client | GraphiQL |
| Prisma | Relay | Grapgh-Playground |
| Grapghql Yoga |  |  |

**AWS service for Grapghql : AWS Appsync**

**Query and Data Modification Operations : 4$ per million query & data modification**

**Real Time updates : $2.00 per million Real-time Updates & $0.08 per million minutes of connection to the AWS AppSync service**

**Caching : Based on instance type**

**GRAPHQL VS REST API**

|  |  |  |
| --- | --- | --- |
| **CONSTRAINT** | **REST API** | **GRAPHQL** |
|  | | |
| **Endpoints** | Multiple  [get/users,post/product,..] | Single  [post /grapghql] |
| **Data** | Over/under fetching | Fetch exactly what we need |
| **Error handling** | Starightforward | Error handling complecated |
| **Cacheable** | Yes | No |
| **Code on demand** | Yes | No |
| **Basic operation** | Get post put delete | Query mutation subscription |
| **Security** | Schema Exposed | Schema not Exposed |
| **Scalabilty** | Yes | No |
| **Portalbility** | Yes | Yes |
| **Consistency** | No | Yes |
| **Ease of server/client development** | No | Yes |
| **Active community** | Limited | Large |
| **Network Performance** | Low | High |
| **Data Speed** | Faster | Slow |
| **Who using it ?** | Netflix, zalando, eyasys | Facebook, twitter, github |

**Get started with Apollo Server**

**1] Create project**

#mkdir graphql-server-example

#cd graphql-server-example

**2] Initialize npm**

#npm init –yes

**3] Install Dependencies**

#npm install apollo-server graphql

**4]** **Define GrapghQl schema, data set & resolver**

#touch index.js



**3] Start the server**

#node index.js