
Ballerina
Swan Lake


# GraphQL API, Apache Kafka and Docker

Dilan Sachintha

October 2023

# In this presentation:

# Introduction

- Ballerina is a modern, open-source, and cloud-native programming language specifically designed for building microservices and distributed systems.
- Has built-in support for creating GraphQL APIs, interacting with Apache Kafka servers and deploy services in cloud.
- Main focus on demonstrating how a Ballerina GraphQL API can be integrated with Apache Kafka and deployed on Docker.

# GraphQL

- A query language for APIs.
- And a runtime for fulfilling those queries with existing data.
- Allows clients to request only the data they need.
- Single endpoint for all data requests.

```
query{
  profile(name: "John"){
    id
    name
    age
  }
}
```

```
{
  "data": {
    "profile": {
      "id": "2",
      "name": "John",
      "age": 20
    }
  }
}
```

# GraphQL with Ballerina

- Ballerina comes with built-in support for creating GraphQL APIs as a part of its network services capabilities.
- Supports both code-first and schema-first approaches.
- Supports all query, mutations and subscriptions without additional libraries.

```
import ballerina/graphql;

Run | Debug | Try it | Visualize
service /graphql on new graphql:Listener(9090) {

    Visualize
    resource function get greeting() returns string {
        return "Hello, World";
    }
}
```

```
type Query {
    greeting: String!
}
```

# Apache Kafka

- Open-source distributed streaming platform.
- Used for building real-time data pipelines and streaming applications.
- Designed to handle high-throughput, fault-tolerant, and scalable data streaming.

# Apache Kafka with Ballerina

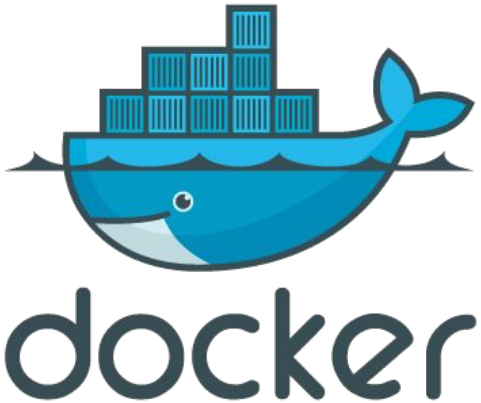
- Ballerina Kafka is a library that provides a set of APIs for interacting with Apache Kafka from Ballerina programs.
- Has clients as Producer, Consumer and Listener.

```
Run | Debug
service on new kafka:Listener(kafka:DEFAULT_URL, {groupId: "order-group-id", topics: "order-topic"}) {

    Visualize
    remote function onConsumerRecord(Order[] orders) {
        from Order 'order in orders
            where 'order.isValid
            do {
                log:printInfo(string `Received valid order for ${'order.productName}`);
            };
    }
}
```

# Docker

- A platform that enables the creation, packaging, and deployment of applications in lightweight, portable containers.
- Containers include all necessary dependencies, ensuring consistent and efficient execution across various environments.
- Widely used for containerized and microservices-based application deployments.



# Docker with Ballerina

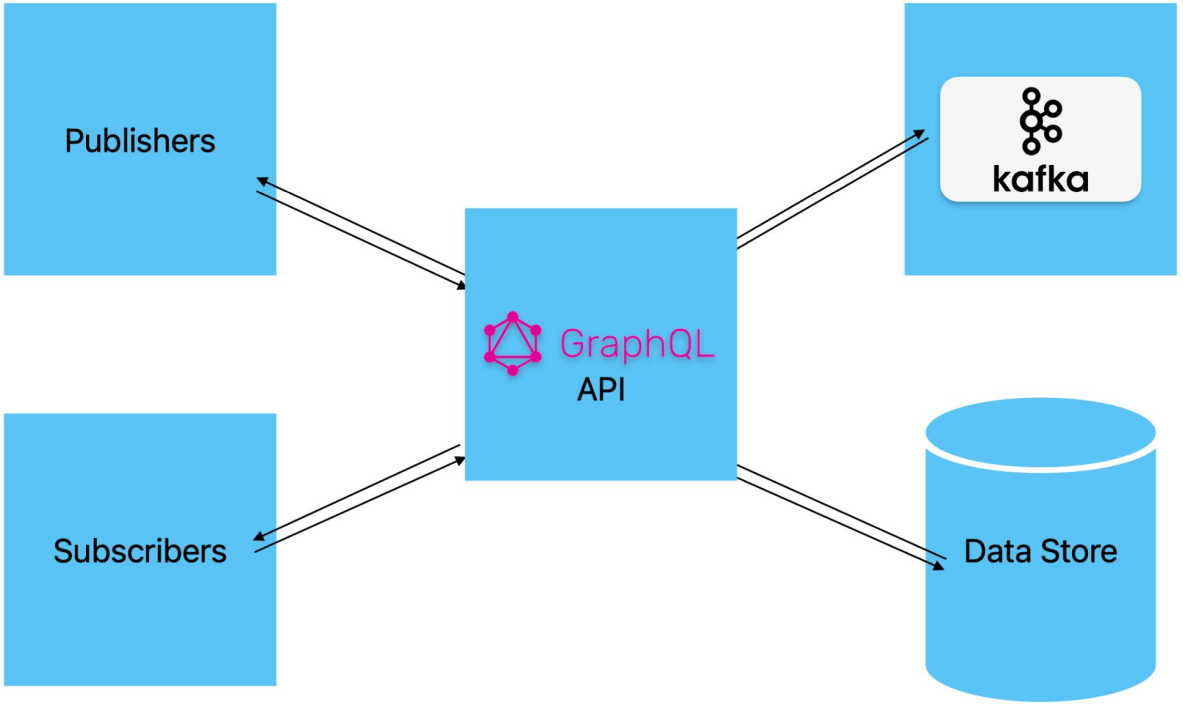
- Ballerina has built in support for creating docker artifacts (C2C).
- C2C is a compiler extension packed with Ballerina.
- Eases generating the artifacts required for the cloud from the Ballerina code.
- Currently has support to generate Docker and Kubernetes artifacts.
- Builds the containers and required artifacts by deriving the required values from the code.
- Generating the docker artifact is as simple as running the command,

```
bal build –cloud=docker
```

# Demonstration

- A sample which showcases
  - How to write a GraphQL service
  - How to use Kafka consumer and producer
  - How to create a docker image for GraphQL service
- Prerequisites
  - Ballerina Swan Lake
  - VS Code, Ballerina VS Code Extension
  - Kafka Cluster
  - Docker

# News Alert System



# Find out more…

- Ballerina documentation
  - Learn guide: GraphQL API with Ballerina
    - https://ballerina.io/learn/write-a-graphql-api-with-ballerina/
  - Learn guide: Code to Cloud deployment in Ballerina
    - https://ballerina.io/learn/code-to-cloud-deployment/
  - Ballerina by example
    - ballerina.io/learn/by-example/

- Join the Ballerina community


ballerinalang


WSO2 Collective

@ballerinalang


ballerina-lang

# Thank you!

If you have any further questions, please email **contact@ballerina.io** or raise them in the **Ballerina Discord server.**