Apache Kafka:

Apache Kafka is a streaming API, which is used to process real data feed with high through put and low latency.

This is developed by LinkedIn and donated to Apache Organization.

It is implemented in SCALA and JAVA programing language.

Apache Kafka is an open source.

Apache Kafka Architecture is based on PUB/SUB.

Real time data: flights data, mobile data, news data, sensors data, stock market…etc.

Apache Kafka terminology:

1. Zookeeper
2. Apache Kafka Server(Message broker)
3. Apache Kafka topic
4. Apache Kafka Producer(Publisher)
5. Apache Kafka Consumer(Subscriber)

* Zookeeper is a server. Which will provide the run time environment to run our Apache Kafka server.
* Kafka server: On top of the zookeeper server we are going to run our apache Kafka server
* Apache Kafka Topic is nothing but a place, where we are going to store our messages in the Kafka server. One Server we can create multiple topics also. Here one topic nothing but one Queue.
* Apache Kafka producer is nothing but an application which is sending the messages to Topic. (publisher)
* Apache Kafka Consumer is nothing but an application which is receiving the messages from Topic. (Subscriber)

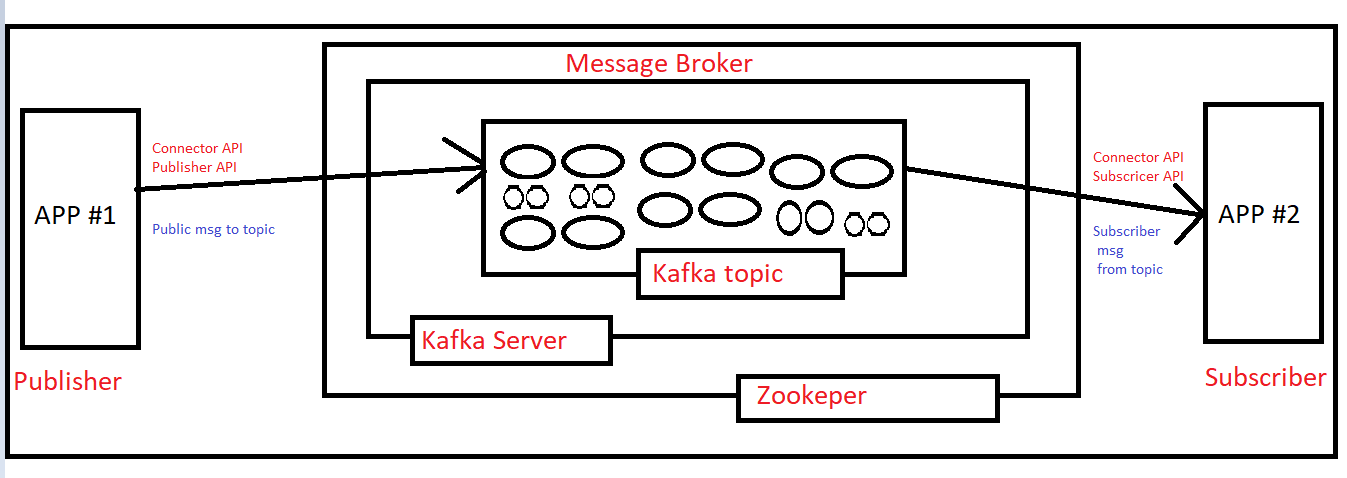
Apache Kafka Internally Uses below 4 API:

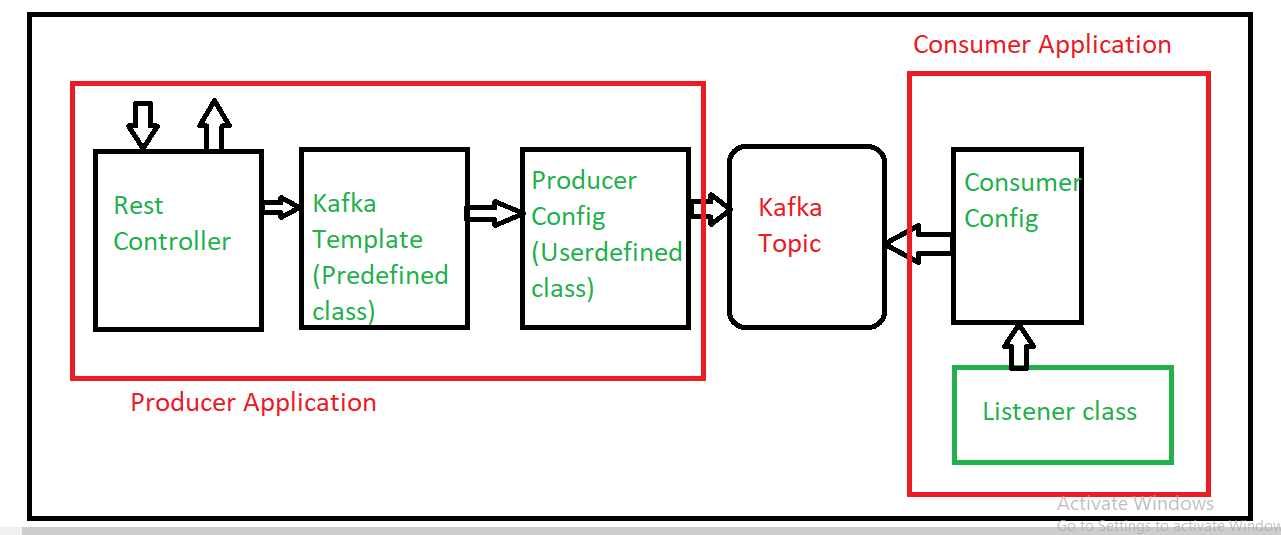
Producer API: It is used to publish messages to Kafka topic

Consumer API: It is used to read the messages from Kafka topic

Connector API: It is used to connect both producer and consumer to Kafka topic.

Stream API: It is used to read the messages from topic and convert them to output result.





Apache Kafka Installation:-

Today's session : Circuit Breaker

---------------------------------

-> Circuit Breaker concept came from Electricals

Wiki-Defintion

---------------

A circuit breaker is an automatically operated electrical switch designed to protect an electrical circuit from damage caused by excess current from an overload or short circuit.

-> This Circuit Breaker concept we can use in our Microservices to implement Fault Tolerance systems.

-> Spring Cloud Netflix library provided Hystrix library to work with Circuit Breaker.

Steps to develop Boot Application with Circuit Breaker

------------------------------------------------------

1) Create Spring Boot Application with below dependencies

1)spring-boot-starter-web

2)spring-cloud-starter-netflix-hystrix

2) Configure @EnableCircuitBreaker in Spring Boot Start Class

3) Create Rest Controller with required methods

4) Configure fallBackMethod using @HystrixCommand annotation to enable CircuitBreaker

5) Configure port number and run boot application.

--------------------------------------------------------------------

@SpringBootApplication

@EnableCircuitBreaker

public class Application {

public static void main(String[] args) {

SpringApplication.run(Application.class, args);

}

}

------------------------------------------------------------------------

@RestController

public class DemoRestController {

@GetMapping("/one")

@HystrixCommand(fallbackMethod = "methodTwo")

public String methodOne() {

String response = "This is from methodOne...!!";

int i = 10 / 0;

return response;

}

public String methodTwo() {

String response = "This is from methodTwo...!!";

return response;

}

}