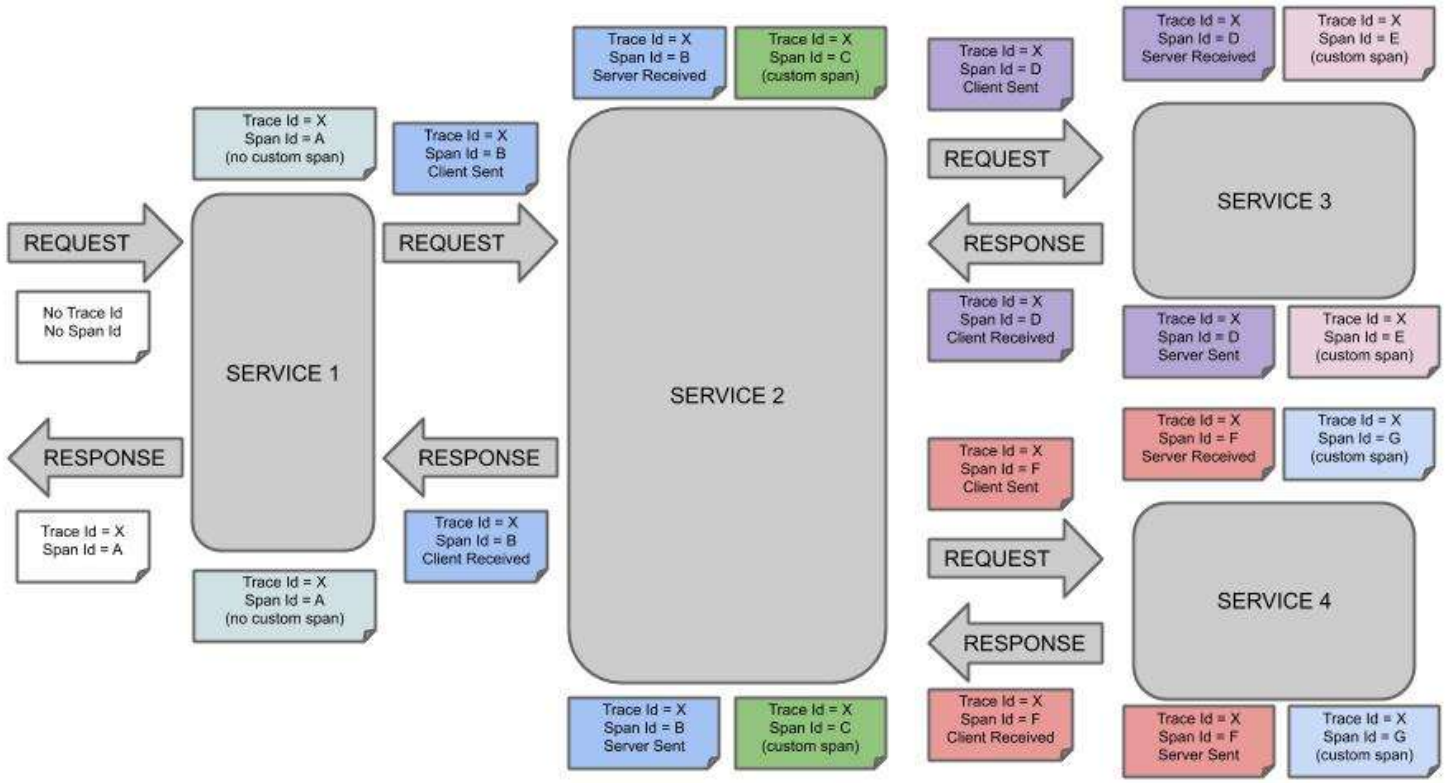
Micrometer Tracing & Zipkin

MSA 환경에서는 여러 서비스가 유기적으로 상호작용

=> 특정 요청이 어떤 서비스들을 거쳐 처리되는지 파악하기 어려움
==> 분산 추적 기술을 통해 파악

분산 추적 기술 : MicrometerTracing & Zipkin

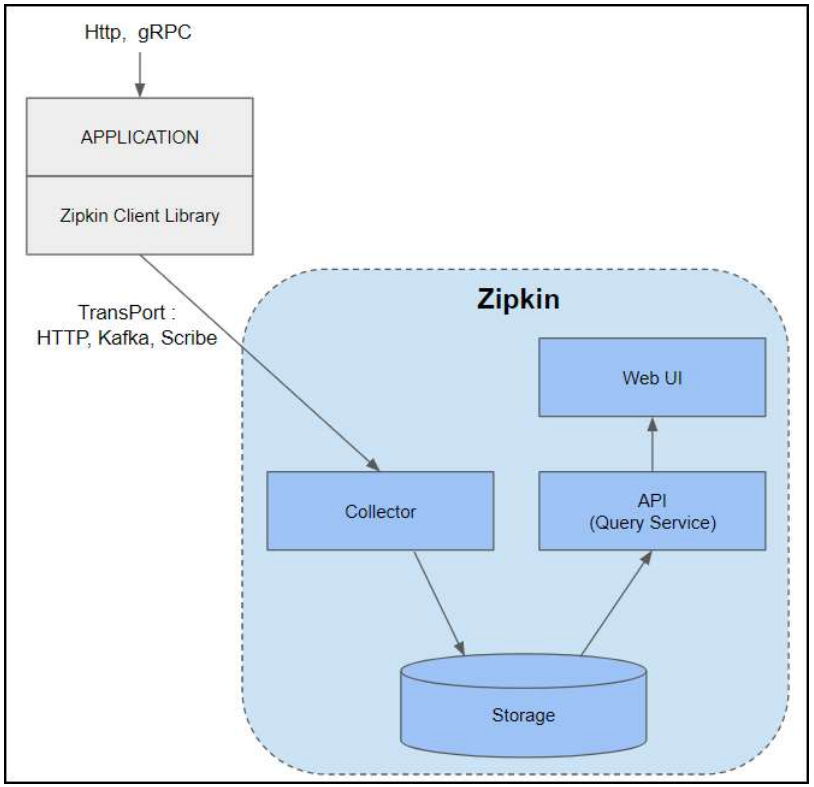
Micrometer Tracing (구 Spring Cloud Sleuth)



스프링 클라우드 생태계에서의 분산 추적을 위한 라이브러리

+ 관찰성 파사드 : 어플리케이션의 가시성을 제공
+ 벤더 중립성 : 특정 모니터링 솔루션이나 기술 스택에 종속되지 않고, 벤더 중립적인 인터페이스 제공
+ 분산 추적 구현 : 요청 흐름 추적을 위한 TraceID & SpanID를 생성하고 전파
+ 컨텍스트 전파 : 서비스 간 통신 시, TraceID & SpanID를 자동으로 다음 서비스로 전달하여 추적 흐름 연결

Zipkin



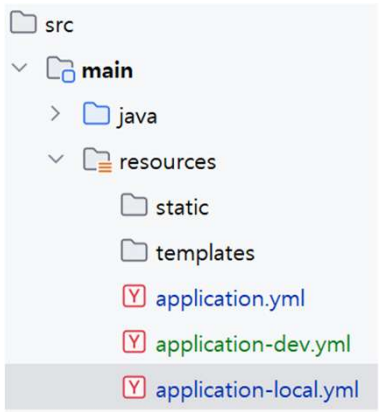
분산 추적 데이터 수집,저장, 시각화하는 `오픈 소스 분산 추적 시스템`

+ 데이터 수집 : Micrometer Tracing과 같은 클라이언트 라이브러리로부터 전송된 추적 데이터(Span) 수집
+ 데이터 저장 : 수집된 추적 데이터 저장 (Using by InMemory, Cassandra, Elasticsearch 등)
+ 데이터 시각화 : 저장된 추적 데이터를 Web UI를 통해 시각화
+ 지연 시간 분석 : 각 Span의 시작 시간과 종료 시간을 기록 -> 지연 시간 분석 및 병목 지점 식별
+ 오류 지점 식별 : 오류가 발생한 지점을 쉽게 찾을 수 있도록 정보 제공 -> 문제 해결 시간 단축

Micrometer Tracing & Zipkin 적용


```
dependencies {
    ...
    implementation 'org.springframework.boot:spring-boot-starter-actuator'
    implementation 'io.micrometer:micrometer-tracing-bridge-otel'
    implementation 'io.opentelemetry:opentelemetry-exporter-zipkin' // Zipkin으로 내보낼 경우
    ...
}
```

Micrometer Tracing & Zipkin 적용 (계속)




```yaml
spring:
  …
  zipkin:
    tracing:
      endpoint: http://localhost:9411/api/v2/spans

management:
  tracing:
    sampling:
      probability: 1.0 # 0.0 ~ 1.0 사이 값 (모든 요청을 트레이싱하려면 1.0)

logging:
  pattern:
    level: "%5p [%X{traceId},%X{spanId}]" # 로그에 traceId, spanId 추가
```

Zipkin 설치 및 실행

노션에 있는 zipkin docker-compose.yml 파일을 아래 디렉토리에 위치

C:₩server₩zipkin>

docker compose up –d 명령어 실행

C:₩server₩zipkin> **docker compose up –d**

http://localhost:9411 접속하여 설치 확인

Micrometer Tracing이 적용된 로그

```
2025-06-04T22:05:09.635+09:00  INFO [3b8c7c77b836be90d08005417bef9c12,db2d8053a57a0d44] 7300 --- [backend-user] [nio-8081-exec-2]
[3b8c7c77b836be90d08005417bef9c12-db2d8053a57a0d44] o.a.k.c.t.i.KafkaMetricsCollector         : initializing Kafka metrics collector
2025-06-04T22:05:09.693+09:00  INFO [3b8c7c77b836be90d08005417bef9c12,db2d8053a57a0d44] 7300 --- [backend-user] [nio-8081-exec-2]
[3b8c7c77b836be90d08005417bef9c12-db2d8053a57a0d44] o.a.k.clients.producer.KafkaProducer     : [Producer clientId=backend-user-producer-1]
...
2025-06-04T22:05:09.804+09:00  INFO [3b8c7c77b836be90d08005417bef9c12,db2d8053a57a0d44] 7300 --- [backend-user] [nio-8081-exec-2]
[3b8c7c77b836be90d08005417bef9c12-db2d8053a57a0d44] o.a.kafka.common.utils.AppInfoParser     : Kafka version: 3.9.1
2025-06-04T22:05:09.804+09:00  INFO [3b8c7c77b836be90d08005417bef9c12,db2d8053a57a0d44] 7300 --- [backend-user] [nio-8081-exec-2]
[3b8c7c77b836be90d08005417bef9c12-db2d8053a57a0d44] o.a.kafka.common.utils.AppInfoParser     : Kafka commitId: f745dfdcee2b9851
2025-06-04T22:05:09.804+09:00  INFO [3b8c7c77b836be90d08005417bef9c12,db2d8053a57a0d44] 7300 --- [backend-user] [nio-8081-exec-2]
[3b8c7c77b836be90d08005417bef9c12-db2d8053a57a0d44] o.a.kafka.common.utils.AppInfoParser     : Kafka startTimeMs: 1749042309801
```

Zipkin Web UI