

시작하기 및 CLI

Moleculer는 Node.js를 위한 빠르고 확장성이 뛰어난 모던 마이크로서비스 프레임워크입니다.

- CLI 설치: `npm install -g moleculer-cli`
- 프로젝트 생성: `moleculer init project my-project`
- 서비스 생성: `moleculer create service user`
- 러너 실행 (개발 모드): `npm run dev` (또는 `moleculer-runner --repl`)
- REPL 대화형 콘솔: `call users.list, services, actions, nodes, emit user.created`

서비스 브로커 (ServiceBroker)

노드의 중앙 제어 관리자 역할을 수행합니다.

```
import { ServiceBroker } from
"moleculer";
```

```
const broker = new ServiceBroker({
  nodeID: "node-1",
  transporter: "NATS", // Redis, MQTT,
  TCP, RabbitMQ 지원
  cacher: "Redis", // Memory, Redis
  캐싱
  serializer: "JSON", // MsgPack,
  ProtoBuf, Avro
  logger: true,
  metrics: true,
  tracing: { enabled: true, exporter:
  "Jaeger" },
});
```

```
broker.start().then(() => {
  broker.call("math.add", { a: 5, b:
  3 }).then(res => console.log(res));
});
```

서비스 정의 및 액션 (Actions)

```
broker.createService({
  name: "math",
  settings: { defaultOffset: 0 },

  actions: {
    add: {
      params: {
        a: { type: "number" },
        b: { type: "number" },

```

```
    },
    cache: { keys: ["a", "b"], ttl:
30 },
    async handler(ctx) {
      return ctx.params.a +
      ctx.params.b;
    },

    // 다른 서비스 호출 예시
    async multiply(ctx) {
      const { a, b } = ctx.params;
      const res = await
      ctx.call("otherService.calc", { a, b });
      return res;
    },

    methods: {
      // 내부 비공개 헬퍼 함수
      formatResult(val) { return `Result:
      ${val}`; }
    }
  });
```

컨텍스트 및 액션 호출 (Context & Calls)

액션 핸들러 내부에서 `ctx` 객체를 통해 메타데이터와 다른 액션에 접근합니다.

- `ctx.params`: 전달받은 파라미터 데이터
- `ctx.meta`: 분산 추적 및 인증 정보가 담긴 메타데이터
- `ctx.call('service.action', params, opts)`: 다른 서비스의 액션 호출
- `ctx.emit('event.name', payload)`: 이벤트 발생
- `ctx.broadcast('event.name', payload)`: 모든 노드로 이벤트 브로드캐스트

이벤트 버스 (Event Bus & Pub/Sub)

```
broker.createService({
  name: "notifications",
  events: {
    // 밸런싱 이벤트 (컨슈머 그룹 중 1개 인스턴스만 수신)

```

```
    "user.created"(ctx) {
      console.log("환영 이메일 발송:",
      ctx.params.email);
    },
    // 브로드캐스트 이벤트 (모든 인스턴스가 수신)
    "$services.changed"(ctx) {
      console.log("서비스 목록 변경 감지");
    }
  });
```

API 게이트웨이 (moleculer-web)

HTTP 요청을 마이크로서비스 액션으로 라우팅합니다.

```
import ApiService from "moleculer-web";

broker.createService({
  mixins: [ApiService],
  settings: {
    port: 3000,
    routes: [
      {
        path: "/api",
        whitelist: ["users.*",
        "posts.*"],
        aliases: {
          "GET users": "users.list",
          "POST users": "users.create",
          "REST posts": "posts", //
          RESTful CRUD 자동 매핑
        },
        autoAliases: true,
      },
    ],
  },
});
```

내결함성 및 장애 복구 (Fault Tolerance)

- 서킷 브레이커 (Circuit Breaker):
`circuitBreaker: { enabled: true, threshold: 0.5, minRequestCount: 20 }`
- 재시도 (Retries): `retryPolicy: { enabled: true, retries: 3, delay: 100 }`

- 타임아웃 (Timeout): `broker.call("service.action", {}, { timeout: 3000 })`
- 벌크헤드 (Bulkhead): 동시 실행 요청 수 제한