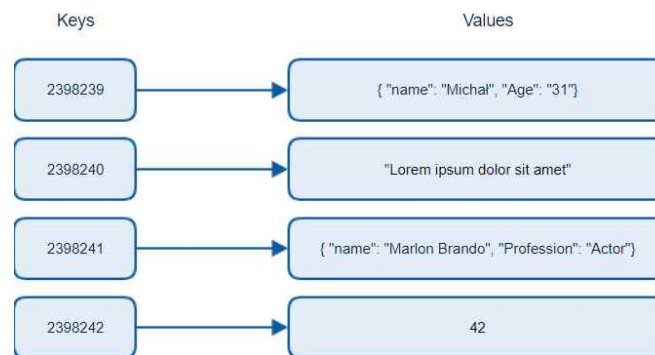


Redis: In-memory Key-value Store

[Redis 소개]

- NoSQL(Not only SQL) 데이터베이스
- 전통적인 관계형 데이터베이스(Oracle, MySQL 등)의 틀에서 벗어난 데이터베이스
- MongoDB, Cassandra, Redis, CouchDB 등
- Redis는 NoSQL DB 중에서도 Key-Value Store: 검색을 위한 Key, 데이터에 해당하는 Value 로 단순한 구성



<그림:

<https://programmerprodigy.code.blog/2021/06/28/introduction-to-key-value-data-store-along-with-use-cases/>

- 다른 DB와 달리 메모리에 데이터를 저장하므로 In-memory NoSQL database 또는 In-memory key-value store라고 부름
- 사용이 쉽고 극단적으로 가볍다는 특징 때문에 서버-클라이언트 구조에서 In-memory cache로 흔히 사용된다.
- Redis 7.2.1 다운로드: <https://redis.io/download/>

[illegible]

[TO DO]

1. redis-server

1.1. aeMain()

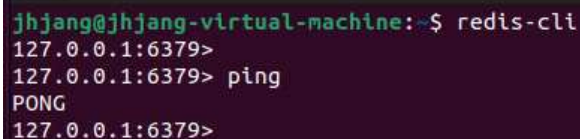
- redis-server는 이벤트 루프의 형식으로 내/외부 요청을 처리한다. ae.c:aeMain() 함수를 중심으로 레디스 서버의 전반적인 동작 과정을 분석하라.
- 처리 과정과 핵심적인 함수 등을 이해하기 쉬운 그림과 함께 설명

1.2. 명령어 처리

- redis-cli에서 SET 명령어를 입력했을 때(예: 'SET x 1'), 명령어가 클라이언트에서 서버로, redis-server로 전달되고, 최종적으로 In-Memory DB에 저장되는 과정을 보여라.
 - * Call graph 작성
 - * 사용되는 자료 구조에 대한 설명
(서버-클라이언트 연결, 사용되는 버퍼, value가 저장되는 형식 등)
 - * 처리 과정을 이해하기 쉬운 그림과 함께 설명

1.3. ping

- ping 명령어는 단순히 "PONG"이라는 문자열을 리턴한다.
(그 외에는 아무 일도 하지 않는다).



```
jhjang@jhjang-virtual-machine:~$ redis-cli
127.0.0.1:6379>
127.0.0.1:6379> ping
PONG
127.0.0.1:6379>
```

- 레디스에 ping 명령어를 추가하시오.
 - * 클라이언트가 ping이라고 이라고 보내면 서버는 "PONG" 문자열을 리턴
 - * 구현 방법 설명
 - * 실행 결과 캡처

2. In-memory Database

2.1. Key-value store

- 레디스 database(key-value store)가 메모리에 저장되는 자료 구조를 분석하라.

2.2. Data types

- <https://redis.io/docs/data-types/>
- Data Type 중 String, List가 어떻게 구현되어 있는지 분석하라.

2.3(Bonus). Binary Search Tree

- 레디스의 Data Type에 이진탐색트리를 추가하라.
- 이진탐색트리에서 데이터를 검색, 삽입하는 'BGET', 'BSET' 명령어를 구현

[제출 방법: hi-class]

- 11월 29일(수요일) 23:59까지, delay x일*10% 감점. 최대 5일
- 보고서(팀장이 제출)
 - * 팀이름, 팀원(학번/이름)
 - * 구현된 소스 코드의 github URL(private으로 작업할 것)