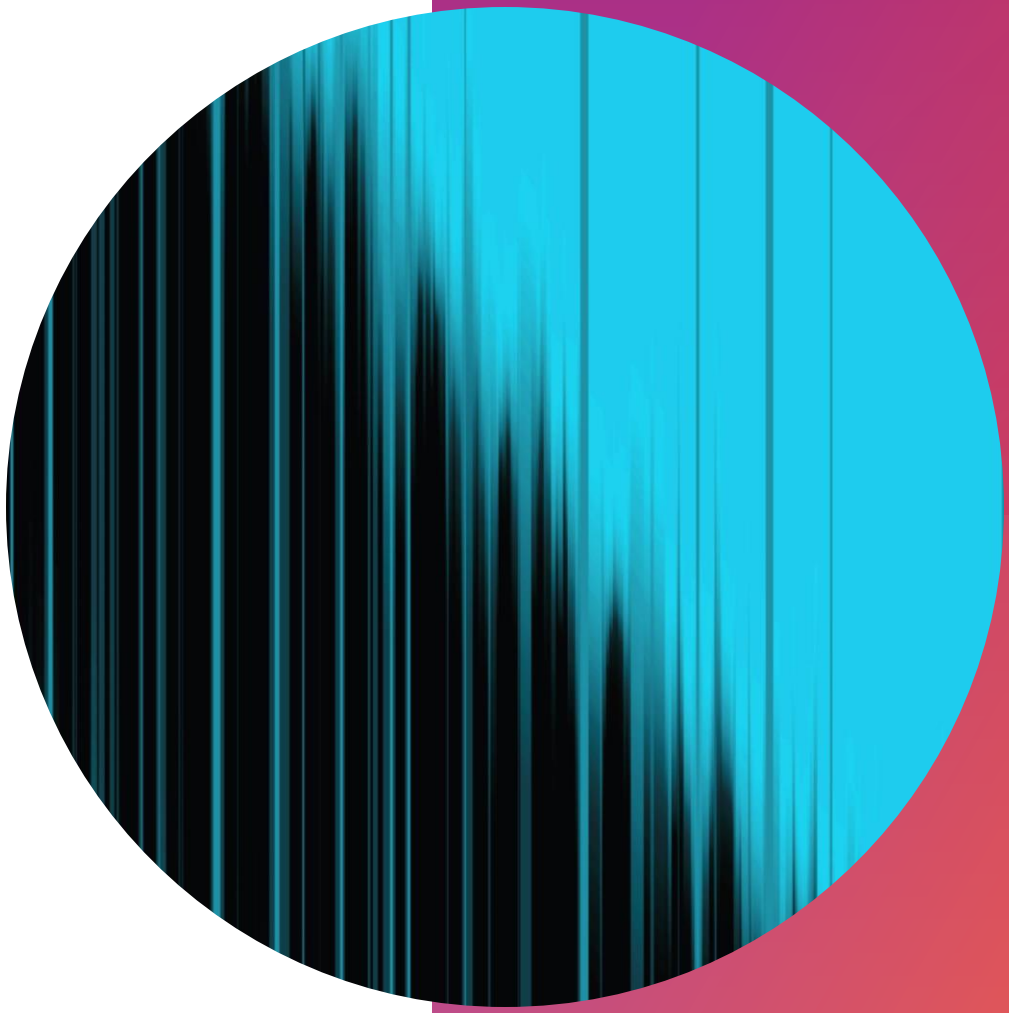
HTTP Daemon



ONEM2M TINY IOT PROJECT

엄경호

이번주 진행 상황

서버의 기본 동작들을 대부분 구현 -> 디테일한 부분을 조금씩 추가할 예정

트리 뷰어 API 구현 및 CORS 문제 해결

DB와 리소스 트리가 가지고 있는 정보 비교

서버 스펙

1. 서버 실행 시 기존 리소스 트리 상태로 재조립

2. OneM2M 헤더 검증, URI 검증

3. 모든 오브젝트에 대하여 기본적인 CR(U)D 동작 -> /latest uri도 유효

4. /viewer uri로 요청이 오면 리소스 트리 API 호출

트리 뷰어 API

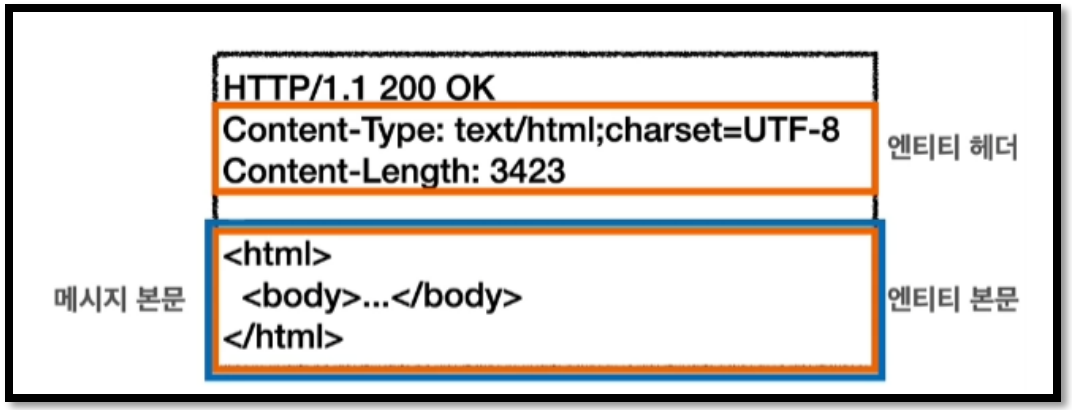
http://host.com/viewer/ObjectURI

CORS 문제 발생 -> 기본적으로 서로 다른 도메인은 리소스를 공유할 수 없지만 이를 허용해주는 정책

이를 해결하기 위해서는 응답 헤더에 Access-Control-Allow-Origin 필드가 존재해야 한다.

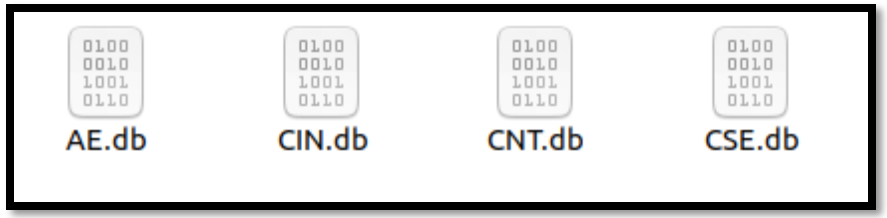
다른 프레임 워크에서는 설정 파일이나 메소드를 통해 헤더를 추가할 수 있지만 순수 C에서는?

트리 뷰어 API



HTTP Entity 작성법

리소스트리 vs DB 비교



```
cbs : -1
cbs : -1
cni : -1
cni : -1
ct : 20220803T060319
ct : 20220803T060321
et : 20220803T060319
et : 20220803T060321
lt : 20220803T060319
lt : 20220803T060321
pi : 2-20220803T060304
pi : 2-20220803T060304
ri : 3-20220803T060319
ri : 3-20220803T060321
rn : test_CNT1
rn : test_CNT2
st : -1
st : -1
ty : 3
ty : 3
```

CNT.db

1. 4개의 파일이 각 1종류의 오브젝트들을 저장함

2. ri를 키 값으로 해당 오브젝트를 Get 가능함

리소스트리 vs DB 비교

```
typedef struct Node{
        struct Node *parent;
        struct Node *child;
        struct Node *siblingLeft;
        struct Node *siblingRight;

        char *rn;
        char *ri;
        char *pi;
        ObjectType ty;
}Node;
```

1. 노드와 오브젝트가 1:1 대응

2. rn으로 URI 요청이 들어오면 rn과 ri를 맵핑함

기본 시나리오

1. URI를 토대로 해당 노드를 탐색

2. rn과 ri를 맵핑 후 DB에 Get 요청

서버 구현 디테일

1. 첫 CIN, 가장 최근 CIN, CIN의 개수를 저장해야 하는데 첫 CIN의 의미 ?

2. CNT와 CIN가 같은 계층에 존재할 수 있나?

첫 CIN의 존재 여부, CNT와 CIN이 같은 계층에 존재하는 지 여부가 디테일한 동작을 구현하는 데에 방향을 결정함!

UPDATE는 이러한 디테일을 모두 구현 후 구현할 예정 -> 독립적인 CRD 동작에 비해 Create와 Delete 구현에 의존적이기 때문