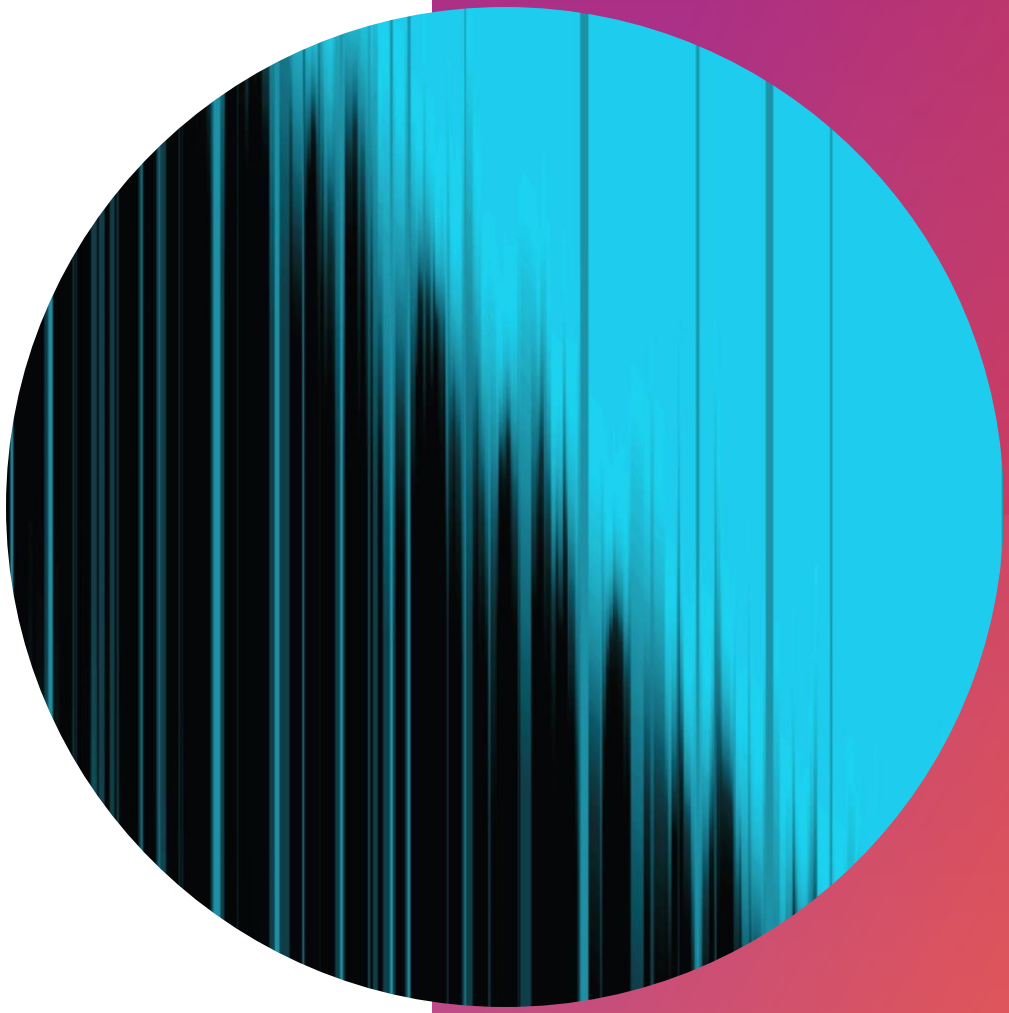
HTTP Daemon



ONEM2M TINY IOT PROJECT

엄경호

저번 주 진행 상황

POST /Mobius HTTP/1.1	
Host:	localhost:7579
X-M2M-Origin:	TAE
X-M2M-RI:	1234
Content-Type:	application/json; ty=2
Accept:	application/json

```
{
    "m2m:ae": {
        "rn": "Sensor1",
        "api": "tinyProject",
        "rr": true
    }
}
```

char* rn
char* api
bool rr

AE 구조체

NULL, 공백, 개행 제거 후 Parsing
By Json Parser

저번 주 진행 상황

char* rn
char* api
bool rr

AE 구조체

구조체를 다시 Json 형태로 재조립
By Json Parser

onem2m.h 설명

onem2m.h에 선언된 내용들을 통해서 앞으로 어떤 식으로 구현해야 HTTPd, json parser, DB가 유기적으로 작동할 수 있는지 설명

onem2m.h 설명

```
typedef enum {
        o_CREATE = 1,
        o_RETRIEVE,
        o_UPDATE,
        o_DELETE
}Operation;

typedef enum {
        t_AE = 2,
        t_CNT,
        t_CIN,
        t_CSE
}ObjectType;
```

op, ty 상수 선언
OneM2M operation, type 정의를 따름

onem2m.h 설명

```c
// OneM2M Resource struct
typedef struct {
        char ct[16];
        char lt[16];
        char *rn;
        char *ri;
        char *pi;
        char *csi;
        int ty;
} CSE;

typedef struct {
        char et[16];
        char ct[16];
        char lt[16];
        char *rn;
        char *ri;
        char *pi;
        char *api;
        char *aei;
        int ty;
        bool rr;
} AE;

typedef struct {
        char et[16];
        char ct[16];
        char lt[16];
        char *rn;
        char *ri;
        char *pi;
        int ty;
        int st;
        int cni;
        int cbs;
} CNT;

typedef struct {
        char et[16];
        char ct[16];
        char lt[16];
        char *rn;
        char *ri;
        char *pi;
        char *csi;
        char *con;
        int ty;
        int st;
} CIN;
```

풀네임	쇼네임	권장 자료형	입력 예시
resourceName	rn	문자열	"Mobius"
resourceID	ri	문자열	"TAE"
resourceType	ty	int	5
parentID	pi	문자열	"5-20191210093452845"
expirationTime	et	char[16]	"20240513T083900"
creationTime	ct	char[16]	"20191210T093452"
lastModifiedTime	lt	char[16]	"20191210T093452"
CSE-ID	csi	문자열	"/Mobius2"
App-ID	api	문자열	"tinyProject"
AE-ID	aei	문자열	"TAE"
requestReachability	rr	문자열	true
stateTag	st	int	0
currentNrOfInstances	cni	int	0
currentByteSize	cbs	int	0
contentSize	cs	int	0
content	con	문자열	"ON"

Full name으로 대략적인 변수의 기능 유추 가능
이 부분에 대해선 나중에 필요하면 자세하게 다룰 예정

onem2m.h 설명

```
// OneM2M Resource function
Operation Parse_Operation();
ObjectType Parse_ObjectType();
ObjectType Parse_ObjectType_By_URI();

AE* Create_AE(char *json_payload);
CNT* Create_CNT(char *json_payload);
CIN* Create_CIN(char *json_payload);

CSE* Retrieve_CSE();
AE* Retrieve_AE();
CNT* Retrieve_CNT();
CIN* Retrieve_CIN();

CSE* Update_CSE(char *json_payload);
AE* Update_AE(char *json_payload);
CNT* Update_CNT(char *json_payload);
CIN* Update_CIN(char *json_payload);

CSE* Delete_CSE();
AE* Delete_AE();
CNT* Delete_CNT();
CIN* Delete_CIN();

CSE* Json_to_CSE(char *json_payload);
AE* Json_to_AE(char *json_payload);
CNT* Json_to_CNT(char *json_payload);
CIN* Json_to_CIN(char *json_payload);

char* CSE_to_json(CSE* cse_object);
char* AE_to_json(AE* ae_object);
char* CNT_to_json(CNT* cnt_object);
char* CIN_to_json(CIN* cin_object);

int Load_CSE(CSE* cse_object);
int Load_AE(AE* ae_object);
int Load_CNT(CNT* cnt_object);
int Load_CIN(CIN* cin_object);

CSE* Get_CSE();
AE* Get_AE();
CNT* Get_CNT();
CIN* Get_CIN();
```

```
Operation Parse_Operation() {
    Operation op

    switch(protocol method) {
    case 'POST': op = o_CREATE
    case 'GET': op = o_RETRIEVE
    case 'PUT': op = o_UPDATE
    case 'DELETE': op = o_DELETE
    }

    return op
}
```

Operation은 프로토콜 바인딩 시 메소드에 바인딩 됨
op 값에 따라 Create, Retrieve, Update, Delete 결정

onem2m.h 설명

```
// OneM2M Resource function
Operation Parse_Operation();
ObjectType Parse_ObjectType();
ObjectType Parse_ObjectType_By_URI();

AE* Create_AE(char *json_payload);
CNT* Create_CNT(char *json_payload);
CIN* Create_CIN(char *json_payload);

CSE* Retrieve_CSE();
AE* Retrieve_AE();
CNT* Retrieve_CNT();
CIN* Retrieve_CIN();

CSE* Update_CSE(char *json_payload);
AE* Update_AE(char *json_payload);
CNT* Update_CNT(char *json_payload);
CIN* Update_CIN(char *json_payload);

CSE* Delete_CSE();
AE* Delete_AE();
CNT* Delete_CNT();
CIN* Delete_CIN();

CSE* Json_to_CSE(char *json_payload);
AE* Json_to_AE(char *json_payload);
CNT* Json_to_CNT(char *json_payload);
CIN* Json_to_CIN(char *json_payload);

char* CSE_to_json(CSE* cse_object);
char* AE_to_json(AE* ae_object);
char* CNT_to_json(CNT* cnt_object);
char* CIN_to_json(CIN* cin_object);

int Load_CSE(CSE* cse_object);
int Load_AE(AE* ae_object);
int Load_CNT(CNT* cnt_object);
int Load_CIN(CIN* cin_object);

CSE* Get_CSE();
AE* Get_AE();
CNT* Get_CNT();
CIN* Get_CIN();
```

```
ObjectType Parse_ObjectType() {
    ObjectType ty

    switch(Content-type헤더의 ty값) {
    case '2': ty = t_AE
    case '3': ty = t_CNT
    case '4': ty = t_CIN
    case '5': ty = t_CSE
    }

    return ty
}
```

Content-Type 헤더 값 -> application/json ty=2;
모든 Operation에 Content-Type 헤더 값을 포함하는건 아님

onem2m.h 설명

```
// OneM2M Resource function
Operation Parse_Operation();
ObjectType Parse_ObjectType();
ObjectType Parse_ObjectType_By_URI();

AE* Create_AE(char *json_payload);
CNT* Create_CNT(char *json_payload);
CIN* Create_CIN(char *json_payload);

CSE* Retrieve_CSE();
AE* Retrieve_AE();
CNT* Retrieve_CNT();
CIN* Retrieve_CIN();

CSE* Update_CSE(char *json_payload);
AE* Update_AE(char *json_payload);
CNT* Update_CNT(char *json_payload);
CIN* Update_CIN(char *json_payload);

CSE* Delete_CSE();
AE* Delete_AE();
CNT* Delete_CNT();
CIN* Delete_CIN();

CSE* Json_to_CSE(char *json_payload);
AE* Json_to_AE(char *json_payload);
CNT* Json_to_CNT(char *json_payload);
CIN* Json_to_CIN(char *json_payload);

char* CSE_to_json(CSE* cse_object);
char* AE_to_json(AE* ae_object);
char* CNT_to_json(CNT* cnt_object);
char* CIN_to_json(CIN* cin_object);

int Load_CSE(CSE* cse_object);
int Load_AE(AE* ae_object);
int Load_CNT(CNT* cnt_object);
int Load_CIN(CIN* cin_object);

CSE* Get_CSE();
AE* Get_AE();
CNT* Get_CNT();
CIN* Get_CIN();
```

```
ObjectType Parse_ObjectType_By_URI() {
    ObjectType ty

    switch(URI의 '/' 개수) {
    case 0: ty = t_CSE
    case 1: ty = t_AE
    case 2: ty = t_CNT
    case 3: ty = t_CIN
    }

    return ty
}
```

OneM2M은 계층 구조임을 활용
이 부분까지는 HTTPd 내에서 구현 가능

onem2m.h 설명

```
// OneM2M Resource function
Operation Parse_Operation();
ObjectType Parse_ObjectType();
ObjectType Parse_ObjectType_By_URI();

AE* Create_AE(char *json_payload);
CNT* Create_CNT(char *json_payload);
CIN* Create_CIN(char *json_payload);

CSE* Retrieve_CSE();
AE* Retrieve_AE();
CNT* Retrieve_CNT();
CIN* Retrieve_CIN();

CSE* Update_CSE(char *json_payload);
AE* Update_AE(char *json_payload);
CNT* Update_CNT(char *json_payload);
CIN* Update_CIN(char *json_payload);

CSE* Delete_CSE();
AE* Delete_AE();
CNT* Delete_CNT();
CIN* Delete_CIN();

CSE* Json_to_CSE(char *json_payload);
AE* Json_to_AE(char *json_payload);
CNT* Json_to_CNT(char *json_payload);
CIN* Json_to_CIN(char *json_payload);

char* CSE_to_json(CSE* cse_object);
char* AE_to_json(AE* ae_object);
char* CNT_to_json(CNT* cnt_object);
char* CIN_to_json(CIN* cin_object);

int Load_CSE(CSE* cse_object);
int Load_AE(AE* ae_object);
int Load_CNT(CNT* cnt_object);
int Load_CIN(CIN* cin_object);

CSE* Get_CSE();
AE* Get_AE();
CNT* Get_CNT();
CIN* Get_CIN();
```

```
int Load_Object(Object* object) {
    Load object to DB // object reference의 값을 참고

    success -> return success value
    fail -> return fail value
}
```

object reference를 parameter로 DB에 해당 내용 적재

onem2m.h 설명

```
// OneM2M Resource function
Operation Parse_Operation();
ObjectType Parse_ObjectType();
ObjectType Parse_ObjectType_By_URI();

AE* Create_AE(char *json_payload);
CNT* Create_CNT(char *json_payload);
CIN* Create_CIN(char *json_payload);

CSE* Retrieve_CSE();
AE* Retrieve_AE();
CNT* Retrieve_CNT();
CIN* Retrieve_CIN();

CSE* Update_CSE(char *json_payload);
AE* Update_AE(char *json_payload);
CNT* Update_CNT(char *json_payload);
CIN* Update_CIN(char *json_payload);

CSE* Delete_CSE();
AE* Delete_AE();
CNT* Delete_CNT();
CIN* Delete_CIN();

CSE* Json_to_CSE(char *json_payload);
AE* Json_to_AE(char *json_payload);
CNT* Json_to_CNT(char *json_payload);
CIN* Json_to_CIN(char *json_payload);

char* CSE_to_json(CSE* cse_object);
char* AE_to_json(AE* ae_object);
char* CNT_to_json(CNT* cnt_object);
char* CIN_to_json(CIN* cin_object);

int Load_CSE(CSE* cse_object);
int Load_AE(AE* ae_object);
int Load_CNT(CNT* cnt_object);
int Load_CIN(CIN* cin_object);

CSE* Get_CSE();
AE* Get_AE();
CNT* Get_CNT();
CIN* Get_CIN();
```

```
Object* Get_Object( ??? ) {
    Object *object = get object from DB // 어떤 정보가 필요한가?

    success -> return object
    fail -> return Null
}
```

DB에서 object를 꺼내오기 위해 필요한 정보가 무엇인지 고민할 필요가 있음

onem2m.h 설명

```
// OneM2M Resource function
Operation Parse_Operation();
ObjectType Parse_ObjectType();
ObjectType Parse_ObjectType_By_URI();

AE* Create_AE(char *json_payload);
CNT* Create_CNT(char *json_payload);
CIN* Create_CIN(char *json_payload);

CSE* Retrieve_CSE();
AE* Retrieve_AE();
CNT* Retrieve_CNT();
CIN* Retrieve_CIN();

CSE* Update_CSE(char *json_payload);
AE* Update_AE(char *json_payload);
CNT* Update_CNT(char *json_payload);
CIN* Update_CIN(char *json_payload);

CSE* Delete_CSE();
AE* Delete_AE();
CNT* Delete_CNT();
CIN* Delete_CIN();

CSE* Json_to_CSE(char *json_payload);
AE* Json_to_AE(char *json_payload);
CNT* Json_to_CNT(char *json_payload);
CIN* Json_to_CIN(char *json_payload);

char* CSE_to_json(CSE* cse_object);
char* AE_to_json(AE* ae_object);
char* CNT_to_json(CNT* cnt_object);
char* CIN_to_json(CIN* cin_object);

int Load_CSE(CSE* cse_object);
int Load_AE(AE* ae_object);
int Load_CNT(CNT* cnt_object);
int Load_CIN(CIN* cin_object);

CSE* Get_CSE();
AE* Get_AE();
CNT* Get_CNT();
CIN* Get_CIN();
```

```
Object* Json_to_Object(char *json_payload) {
    Object *object = create object by cJson // json_payload  값 참고

    return object
}
```

json_payload 값을 토대로 object 생성

onem2m.h 설명

```
// OneM2M Resource function
Operation Parse_Operation();
ObjectType Parse_ObjectType();
ObjectType Parse_ObjectType_By_URI();

AE* Create_AE(char *json_payload);
CNT* Create_CNT(char *json_payload);
CIN* Create_CIN(char *json_payload);

CSE* Retrieve_CSE();
AE* Retrieve_AE();
CNT* Retrieve_CNT();
CIN* Retrieve_CIN();

CSE* Update_CSE(char *json_payload);
AE* Update_AE(char *json_payload);
CNT* Update_CNT(char *json_payload);
CIN* Update_CIN(char *json_payload);

CSE* Delete_CSE();
AE* Delete_AE();
CNT* Delete_CNT();
CIN* Delete_CIN();

CSE* Json_to_CSE(char *json_payload);
AE* Json_to_AE(char *json_payload);
CNT* Json_to_CNT(char *json_payload);
CIN* Json_to_CIN(char *json_payload);

char* CSE_to_json(CSE* cse_object);
char* AE_to_json(AE* ae_object);
char* CNT_to_json(CNT* cnt_object);
char* CIN_to_json(CIN* cin_object);

int Load_CSE(CSE* cse_object);
int Load_AE(AE* ae_object);
int Load_CNT(CNT* cnt_object);
int Load_CIN(CIN* cin_object);

CSE* Get_CSE();
AE* Get_AE();
CNT* Get_CNT();
CIN* Get_CIN();
```

```
char* Object_to_json(Object *object) {
    char *json = create json by cJson // object 값 참고

    return json
}
```

object 값을 토대로 json 생성

onem2m.h 설명

```
// OneM2M Resource function
Operation Parse_Operation();
ObjectType Parse_ObjectType();
ObjectType Parse_ObjectType_By_URI();

AE* Create_AE(char *json_payload);
CNT* Create_CNT(char *json_payload);
CIN* Create_CIN(char *json_payload);

CSE* Retrieve_CSE();
AE* Retrieve_AE();
CNT* Retrieve_CNT();
CIN* Retrieve_CIN();

CSE* Update_CSE(char *json_payload);
AE* Update_AE(char *json_payload);
CNT* Update_CNT(char *json_payload);
CIN* Update_CIN(char *json_payload);

CSE* Delete_CSE();
AE* Delete_AE();
CNT* Delete_CNT();
CIN* Delete_CIN();

CSE* Json_to_CSE(char *json_payload);
AE* Json_to_AE(char *json_payload);
CNT* Json_to_CNT(char *json_payload);
CIN* Json_to_CIN(char *json_payload);

char* CSE_to_json(CSE* cse_object);
char* AE_to_json(AE* ae_object);
char* CNT_to_json(CNT* cnt_object);
char* CIN_to_json(CIN* cin_object);

int Load_CSE(CSE* cse_object);
int Load_AE(AE* ae_object);
int Load_CNT(CNT* cnt_object);
int Load_CIN(CIN* cin_object);

CSE* Get_CSE();
AE* Get_AE();
CNT* Get_CNT();
CIN* Get_CIN();
```

```
Object* Create_Object(char *json_payload) {
    Object *object = Json_to_Object(json_payload)

    int result = Load_Object(object)

    If(result == success) HTTP_201 with Object_to_json(object)
    else HTTP_500

    return object
}
```

onem2m.h 설명

```
// OneM2M Resource function
Operation Parse_Operation();
ObjectType Parse_ObjectType();
ObjectType Parse_ObjectType_By_URI();

AE* Create_AE(char *json_payload);
CNT* Create_CNT(char *json_payload);
CIN* Create_CIN(char *json_payload);

CSE* Retrieve_CSE();
AE* Retrieve_AE();
CNT* Retrieve_CNT();
CIN* Retrieve_CIN();

CSE* Update_CSE(char *json_payload);
AE* Update_AE(char *json_payload);
CNT* Update_CNT(char *json_payload);
CIN* Update_CIN(char *json_payload);

CSE* Delete_CSE();
AE* Delete_AE();
CNT* Delete_CNT();
CIN* Delete_CIN();

CSE* Json_to_CSE(char *json_payload);
AE* Json_to_AE(char *json_payload);
CNT* Json_to_CNT(char *json_payload);
CIN* Json_to_CIN(char *json_payload);

char* CSE_to_json(CSE* cse_object);
char* AE_to_json(AE* ae_object);
char* CNT_to_json(CNT* cnt_object);
char* CIN_to_json(CIN* cin_object);

int Load_CSE(CSE* cse_object);
int Load_AE(AE* ae_object);
int Load_CNT(CNT* cnt_object);
int Load_CIN(CIN* cin_object);

CSE* Get_CSE();
AE* Get_AE();
CNT* Get_CNT();
CIN* Get_CIN();
```

```
Object* Retrieve_Object( ??? ) {
    Object *object = Get_Object( ??? )

    If(object) HTTP_200 with Object_to_json(object)
    else HTTP_500

    return object
}
```

onem2m.h 설명

```
// OneM2M Resource function
Operation Parse_Operation();
ObjectType Parse_ObjectType();
ObjectType Parse_ObjectType_By_URI();

AE* Create_AE(char *json_payload);
CNT* Create_CNT(char *json_payload);
CIN* Create_CIN(char *json_payload);

CSE* Retrieve_CSE();
AE* Retrieve_AE();
CNT* Retrieve_CNT();
CIN* Retrieve_CIN();

CSE* Update_CSE(char *json_payload);
AE* Update_AE(char *json_payload);
CNT* Update_CNT(char *json_payload);
CIN* Update_CIN(char *json_payload);

CSE* Delete_CSE();
AE* Delete_AE();
CNT* Delete_CNT();
CIN* Delete_CIN();

CSE* Json_to_CSE(char *json_payload);
AE* Json_to_AE(char *json_payload);
CNT* Json_to_CNT(char *json_payload);
CIN* Json_to_CIN(char *json_payload);

char* CSE_to_json(CSE* cse_object);
char* AE_to_json(AE* ae_object);
char* CNT_to_json(CNT* cnt_object);
char* CIN_to_json(CIN* cin_object);

int Load_CSE(CSE* cse_object);
int Load_AE(AE* ae_object);
int Load_CNT(CNT* cnt_object);
int Load_CIN(CIN* cin_object);

CSE* Get_CSE();
AE* Get_AE();
CNT* Get_CNT();
CIN* Get_CIN();
```

```
Object* Update_Object(char *json_payload) {
    Object *object = Json_to_Object(json_payload)

    ?????

    return object
}
```

1. Delete 후 Create
2. Get 후 Edit

onem2m.h 설명

```
// OneM2M Resource function
Operation Parse_Operation();
ObjectType Parse_ObjectType();
ObjectType Parse_ObjectType_By_URI();

AE* Create_AE(char *json_payload);
CNT* Create_CNT(char *json_payload);
CIN* Create_CIN(char *json_payload);

CSE* Retrieve_CSE();
AE* Retrieve_AE();
CNT* Retrieve_CNT();
CIN* Retrieve_CIN();

CSE* Update_CSE(char *json_payload);
AE* Update_AE(char *json_payload);
CNT* Update_CNT(char *json_payload);
CIN* Update_CIN(char *json_payload);

CSE* Delete_CSE();
AE* Delete_AE();
CNT* Delete_CNT();
CIN* Delete_CIN();

CSE* Json_to_CSE(char *json_payload);
AE* Json_to_AE(char *json_payload);
CNT* Json_to_CNT(char *json_payload);
CIN* Json_to_CIN(char *json_payload);

char* CSE_to_json(CSE* cse_object);
char* AE_to_json(AE* ae_object);
char* CNT_to_json(CNT* cnt_object);
char* CIN_to_json(CIN* cin_object);

int Load_CSE(CSE* cse_object);
int Load_AE(AE* ae_object);
int Load_CNT(CNT* cnt_object);
int Load_CIN(CIN* cin_object);

CSE* Get_CSE();
AE* Get_AE();
CNT* Get_CNT();
CIN* Get_CIN();
```

```
Object* Delete_Object( ???? ) {
    Object *object = Get_Object( ??? )

    If(object) Delete Object on DB, HTTP_200 with Object_to_json(object)
    else HTTP_500

    return object
}
```

OneM2M URI 처리 관련

POST('/censor') { // 이러한 URI로 POST 요청이 오면

~~~~~~~~~~~~~~~~~~~~~~~~~~~~ 한 내용을 처리한다.

}

하드 코딩된 처리 방식은 OneM2M에서는 적합하지 않음
~~~~~~~~~~~~~~~~~~~~~~~~~~~~

OneM2M URI 처리 관련

예시

1. http://x.x.x.x/ 로 "abc" AE create 요청

2. http://x.x.x.x/abc "def" CNT create 요청 -> 처리 불가

Operation과 ObjectType 파싱 과정이 필요함

구상도