# 1) 라우터 기본

### GET `/`

- **설명**: 라우터 정보 확인
- **성공(200)**: `{ r: "ok", info: "mission manager (tcp admin api)" }`
  router

### GET `/state`

- **설명**: TCP 서버 상태/요약 반환
- **응답**: `{ r:"ok", state, metadata_keys, banks }`
  - `state`: `{ ip, port, timeoutMs, version, clients, banks }` (tcp.state 그대로 전달)
  - `metadata_keys`: `metadataJson`의 최상위 키 개수
  - `banks`: `state.banks` 값
    router

# 2) 메타데이터( metadataJson ) 관리

### GET `/metadata`

- **설명**: `metadataJson` 전체 스냅샷 복제본 반환
  router

### GET `/metadata/keys?flat=0|1`

- **설명**: 키 목록
- `flat=0`(기본): 최상위 키 목록

- `flat=1`: `a.b.c` 형태로 모든 leaf 경로 평탄화 목록
router

## GET `/metadata/:path`

- **설명**: 점 표기 경로(`pos.x.y` 등)로 값 조회
router

## POST `/metadata`

- **설명**: 전체 교체(덮어쓰기)
- **Body**: JSON 객체만 허용
- **성공**: `{ r:"ok", msg:"replaced", size:<키수> }`
router

## PATCH `/metadata`

- **설명**: 부분 병합(딥 머지)
- **Body**: JSON 객체만 허용
- **성공**: `{ r:"ok", msg:"merged" }`
router

## DELETE `/metadata`

- **설명**: 전체 초기화(빈 객체로)
- **성공**: `{ r:"ok", msg:"cleared" }`
router

## PUT `/metadata/:path`

- **설명**: 경로 단위 Upsert (통째 교체)
- **Body**:
  - `{ "value": any }` **또는** 임의 JSON(Object/Array/Primitive) 원본

- **성공**: `{ r:"ok", path, replaced:true }`
router

### PATCH `/metadata/:path`

- **설명**: 경로 단위 부분 병합
    - 대상이 **object & object** 이면 deepMerge
    - 그 외엔 통째 교체
- **성공**: `{ r:"ok", path, merged:true }` **또는** `{ r:"ok", path, replaced:true }`
router

### DELETE `/metadata/:path`

- **설명**: 경로 단위 삭제
- **성공**: `{ r:"ok", path, deleted:<bool> }`
router

### POST `/metadata/save`

- **설명**: 서버 측 메타데이터 파일 저장 (`tcp.saveMetadata()`)
- **성공**: `{ r:"ok", saved:true, path:"..." }` (실패 시 `r:"err"`)
router

### POST `/metadata/load`

- **설명**: 서버 측 파일에서 메타데이터 로드 (`tcp.loadMetadata()`)
- **성공**: `{ r:"ok", loaded:true, path:"..." }` (실패 시 `r:"err"`)
router

## 3) 이미지 뱅크 ( imageBank )

### GET `/banks`

- **설명**: 모든 뱅크 메타 목록

**성공**:

```
{ "r":"ok", "banks":[
  { "bank_id":1, "img_type":0|1|2, "img_size":12345, "img_seq":1,
"ts":1730960000 }
] }
``` :contentReference[oaicite:18]{index=18}
```

- 

### GET `/banks/:bankId?[download=1]`

- **설명**: 메타 조회(기본) 또는 **바이너리 다운로드**(`download=1`)

- `download=1`이면 `Content-Type`은 이미지 타입에 맞춰 설정(JPEG/PNG/BMP), `Content-Length` 포함, 본문은 원본 바이너리.

- 존재하지 않으면 `{ r:"ok", exists:false, bank_id }` 반환.
router

### PUT `/banks/:bankId`

- **설명**: 이미지 업로드 (바디는 `application/octet-stream`)

- **Headers**:

    - `x-img-type`: **필수**. `0(JPG) | 1(PNG) | 2(BMP)`

    - `x-img-seq`: 선택, 기본 `1`

- **성공**: `{ r:"ok", bank_id, size:<bytes> }`

- **검증 실패**: `400` (예: 잘못된 bankId, 잘못된 x-img-type)
router

### DELETE `/banks/:bankId`

- **설명**: 해당 뱅크 삭제

- **성공**: `{ r:"ok", deleted:true|false, bank_id }`
router

## 4) 브로드캐스트/유틸
```

### POST `/broadcast`

- **설명**: 연결된 모든 TCP 클라이언트에 JSON push
- **Body**: 임의의 JSON 객체(배열/원시 X → 객체만)
- **성공**: `{ r:"ok", pushed:true }`
  router

### POST `/ping`

- **설명**: `{cmd:"ping", server_time:<epochSec>}` 을 브로드캐스트
- **성공**: `{ r:"ok" }`
  router

---

# 요청/응답 예시

아래 예시의 Base URL은 `/api/v1/mms` 가정

### 1) 메타데이터 전체 조회

```
curl -s http://localhost:8080/api/v1/mms/metadata
```

### 2) 메타데이터 전체 교체

```
curl -s -X POST http://localhost:8080/api/v1/mms/metadata \
  -H "Content-Type: application/json" \
  -d '{"robot":{"pos":{"x":12,"y":34},"mode":"auto"}}'
```

### 3) 경로 단위 Upsert (교체)

```
curl -s -X PUT http://localhost:8080/api/v1/mms/metadata/robot.pos \
  -H "Content-Type: application/json" \
  -d '{"x":100,"y":200}'
# 또는 { "value": ... } 래퍼 형식도 허용
```

### 4) 경로 단위 부분 병합

```
curl -s -X PATCH http://localhost:8080/api/v1/mms/metadata/robot.pos \
  -H "Content-Type: application/json" \
  -d '{"x":150}'
```

### 5) 이미지 업로드 (PNG)

```
curl -s -X PUT http://localhost:8080/api/v1/mms/banks/1 \
  -H "x-img-type: 1" -H "x-img-seq: 3" \
  -H "Content-Type: application/octet-stream" \
  --data-binary @/path/to/image.png
```

### 6) 이미지 다운로드

```
curl -s -L "http://localhost:8080/api/v1/mms/banks/1?download=1" -o out.png
```

### 7) 브로드캐스트

```
curl -s -X POST http://localhost:8080/api/v1/mms/broadcast \
  -H "Content-Type: application/json" \
  -d '{"cmd":"hello","payload":{"a":1}}'
```