

게이트웨이 안내서

AnyAPI 1.0

TMAXSOFT

저작권 공지

Copyright 2025. TmaxSoft Co., Ltd. All Rights Reserved.

회사 정보

(주)티맥스소프트

주소 : 경기도 성남시 분당구 황새울로258번길 29, 티맥스수내타워 8-9층

기술 서비스 센터: 1544-8629

홈페이지: <https://www.tmaxsoft.com>

제한된 권리

이 소프트웨어(AnyAPI™) 사용설명서와 프로그램은 저작권법과 국제 조약에 의해 보호됩니다. 사용설명서와 프로그램은 TmaxSoft Co., Ltd.와의 사용권 계약 하에서만 사용할 수 있으며, 사용설명서는 사용권 계약의 범위 내에서만 배포 또는 복제할 수 있습니다. 이 사용설명서의 전부 또는 일부를 TmaxSoft의 사전 서면 동의 없이 전자, 기계, 녹음 등의 수단으로 전송, 복제, 배포하거나 2차적 저작물을 작성할 수 없습니다.

이 소프트웨어 사용설명서와 프로그램의 사용권 계약은 어떠한 경우에도 사용설명서 및 프로그램과 관련된 지적 재산권(등록 여부를 불문)을 양도하는 것으로 해석되지 않으며, 브랜드나 로고, 상표 등을 사용할 권한을 부여하지 않습니다. 사용설명서는 오로지 정보 제공만을 목적으로 하며, 이로 인한 계약상의 직접적 또는 간접적 책임을 지지 않습니다. 또한 사용설명서 상의 내용이 법적 또는 상업적인 특정 조건을 만족시킬 것을 보장하지 않습니다. 사용설명서는 제품의 업그레이드나 수정에 따라 예고 없이 변경될 수 있으며, 내용상의 오류가 없음을 보장하지 않습니다.

상표 공지

AnyAPI™는 TmaxSoft Co., Ltd.의 상표입니다. 본 사용설명서에 기재된 모든 제품과 회사 이름은 각각 해당 소유주의 상표로서 참조용으로만 사용되며 반드시 상표 표시(™, ®)를 하지는 않습니다.

Noto는 Google Inc.의 상표입니다. Noto 글꼴은 오픈 소스입니다. 모든 Noto 글꼴은 SIL Open Font License, 버전 1.1에 따라 게시됩니다. (<https://www.google.com/get/noto/>)

오픈소스 소프트웨어 공지

본 제품의 일부 파일 또는 모듈은 다음의 라이선스를 준수합니다. : MIT, LGPL, PSFL

관련 상세 정보는 제품의 다음 디렉터리에 기재된 사항을 참고하시기 바랍니다. : `${INSTALL_PATH}/lib/licenses`

유지 보수

구분	지원항목	서비스 내용
제품지원	패치 & 업그레이드	무상 패치 서비스 제공
		메이저 버전 업그레이드 시 할인 혜택
		웹 지원을 통한 패치 내역 제공
기술 지원 - 기본 서비스	장애 지원	장애 발생 시 원인 분석 및 조치 Service Desk팀 → 기술팀 → R&D의 3단계 장애 분석 및 조치
	일상 지원(온라인 지원)	E-mail, 전화, 원격, 웹 사이트 등 온라인 자원을 통한 질의 응답 서비스
	고객 맞춤 지원(방문 지원)	고객의 요청으로 수행하는 방문 지원 서비스
기술 지원 - 옵션 서비스	예방 지원	정기 점검을 통한 시스템 운영현황 보고 및 장애 예방 ◦ 관리자 또는 운영자의 요구사항 수렴 ◦ 운영 현황(시스템, 엔진 운영) 보고서 제공 ◦ 필요 시 시스템 개선 권장 사항 보고
유지 보수 비용 및 기간	계약 시 별도 협의	계약 시 EOL/EOS 문서 제공

안내서 이력

제품 버전	안내서 버전	발행일	비고
AnyAPI 1.0	3.1.2	2025-01-02	-
AnyAPI 1.0	3.1.1	2024-08-30	-

목차

1. 소개	1
1.1. 개요	1
1.2. 특징	1
1.3. 게이트웨이 환경	2
1.3.1. 게이트웨이 설정 파일	2
1.3.2. 디렉터리 구조	12
1.3.3. 환경 변수	13
2. 엔드포인트 구성	15
2.1. 개요	15
2.1.1. 주요 특징	15
2.1.2. AnyAPI 기본 엔드포인트	15
2.2. 엔드포인트 설정	16
부록 A: 문제 해결	19
A.1. FailedGatewayReadsizeBelowZero	19
A.2. GatewayAPIAuthenticationFailException	19
A.3. GatewayAPINoMatchingConditionException	19
A.4. GatewayEndpointNotExistException	19
A.5. GatewaySslInitFailException	19
A.6. SSL unknown error	20
A.7. Invalid HTTP method	20

1. 소개

본 장에서는 AnyAPI 게이트웨이의 특징 및 설정 파일의 설정 방법에 대해 설명합니다.

1.1. 개요

AnyAPI 게이트웨이는 클라이언트의 API 요청을 AnyAPI 게이트웨이에 등록된 라우팅 정보에 따라 각 API 서버들로 전달하고, 각 API 서버들로부터 받은 응답들을 취합하여 클라이언트에 전달합니다.

또한 라우팅 정보를 통하여 요청 URI 검사, 파라미터 검사 등을 수행합니다. 여러 검사 결과에 따라 라우팅할 Backend API 서버의 URI가 특정되면 해당 정보를 마스터로 전달하여 인증/인가, Qos 처리를 추가로 진행합니다. 이후 다시 게이트웨이를 통하여 실제 API 서버로 라우팅을 한 후 응답을 돌려 받습니다.

1.2. 특징

AnyAPI 게이트웨이의 특징은 다음과 같습니다.

- **Non-Blocking 방식**

I/O 멀티플렉싱 방식의 입출력 처리를 하며, 특정 I/O 채널의 버퍼링에 다른 채널 처리가 영향받지 않도록 fully non-blocking 방식의 처리를 구현하고 있습니다. 기본적으로 1개의 프로세스로 이루어져 하나의 이벤트 스레드에서 non blocking I/O를 통해 HTTP 프로토콜을 처리하며 데이터에 대한 암호화 처리, blocking I/O가 필요한 경우 워커 스레드를 통해 처리합니다.

- **HTTP 프로토콜 지원**

HTTP 버전으로 1.0, 1.1 두 종류를 지원하며, HTTPS 프로토콜을 지원합니다. 또한 HTTP로 들어온 요청을 HTTPS로 변환하는 기능도 지원합니다.

- **라우팅 및 변환**

HTTP 요청이 들어오면 Path나 Method를 기반으로 API를 식별하는 라우팅을 지원합니다. Path에서 와일드카드(*)를 사용하여 모든 문자열에 대응할 수 있습니다. 또한 Path나 Method를 다른 Path, Method로 변환을 지원합니다. Path의 경우 특정 예약어를 통해 지정한 문자열로 대체할 수 있습니다.

항목	설명
{FullPath}	들어온 요청의 전체 Path로 치환합니다.
{StagePath}	Stage의 basePath로 치환합니다.
{Path}	들어온 요청의 전체 Path에서 stage basePath를 제외한 Api Path로 치환합니다.

- **목적지 설정**

서버나 서버 그룹 또는 IP를 직접 지정하여 해당 경로로 프록시 또는 리다이렉트 응답을 돌려주는 기능을 지원합니다. 서버 그룹을 설정하여 라운드로빈 방식, Weight 방식의 목적지 라우팅을 설정할 수 있습니다.

• API 배포 및 테스트 기능

구현한 API는 배포 버튼을 누르기 전까지 외부에 노출되지 않습니다. 배포하지 않은 API는 AnyAPI의 테스트 기능을 통해 정상적으로 동작하는지, API 서버가 어떤 응답을 돌려주는지, API 서버로부터 응답이 얼마나 걸리는지를 확인할 수 있습니다.

• 인증/인가

인증된 사용자를 확인하는 인증 기능과 API 호출 횟수를 제한할 수 있는 OQS 인가 기능을 제공합니다. QOS는 STAGE별로 설정이 가능하며 분당, 시간당, 하루당 얼마큼 처리할 것인지 지정할 수 있습니다.

다음은 AnyAPI 게이트웨이에서 현재 제공하는 인증 방식의 종류입니다.

구분	설명
API Key 인증	HTTP 헤더에 들어온 x-api-key 헤더를 확인하여 등록된 API Key인지 확인하여 사용자를 인증하는 방식입니다.
OAuth 인증	OAuth 서버로부터 접근 토큰을 발급받아 Authorization 헤더에 넣어서 요청하면 AnyAPI 게이트웨이는 Authorization 토큰이 유효한지 체크하여 사용자를 인증하는 방식입니다.
Terminal 인증	TerminalLogin API를 호출하여 단말 토큰을 생성한 후 x-api-token, x-api-pk-number 헤더에 넣어서 요청하면 AnyAPI 게이트웨이는 해당 토큰이 유효한지 체크하여 사용자를 인증하는 방식입니다.

1.3. 게이트웨이 환경

1.3.1. 게이트웨이 설정 파일

AnyAPI 게이트웨이에서 사용하는 설정 파일은 초기 게이트웨이 설정과 동적으로 반영되는 게이트웨이 설정을 저장하기 위해 사용합니다.

설치 초기에 설정 파일은 다음과 같은 위치에 존재합니다.

```
`${GATEWAY_HOME}/config/defaultApiGateway21
```

다음은 AnyAPI 게이트웨이 설정 파일의 작성 예시입니다. (현재 JSON 형식만 지원)



IP, PORT, Monitoring 설정을 제외하고는 사용자가 직접 수정할 수 없으며, WebAdmin을 통해서만 수정이 가능합니다.

• gatewayConfig

게이트웨이에 대한 기본 메타 데이터 설정 정보입니다.

```

"gatewayConfig": {
  "gatewayGroupId": "${GROUP_ID}", (1)
  "id": "${ID}", (2)
  "name": "${NAME}", (3)
  "protocol": "HTTP" (4)
  "thread_monitoring": 1000 (5)
  "thread_pool": { (6)
    "min": 10,
    "max": 50
  }
}

```

각 설정 항목에 대한 설명은 다음과 같습니다.

항목	설명
(1) gatewayGroupId	게이트웨이가 배포되는 그룹의 ID입니다.
(2) id	게이트웨이의 고유 ID입니다.
(3) name	게이트웨이의 이름입니다. 설정하지 않는 경우 ID를 사용합니다.
(4) protocol	게이트웨이가 사용하는 프로토콜입니다. (기본값: HTTP)
(5) thread_monitoring	스레드 모니터링용 로그의 주기(ms)를 설정합니다. (기본값: 1000)
(6) thread_pool	<p>기본 스레드 풀을 설정합니다.</p> <p>실제 파싱이나 매핑이 이루어질 때 사용됩니다. 이때 스레드 풀은 선형적으로 증가하지는 않으며, 상황에 맞게 max 값까지 증가합니다.</p> <ul style="list-style-type: none"> ◦ min: 부팅 시 최소로 뜨는 스레드 개수 ◦ max: 최대로 늘어나는 스레드 개수

• staticResources

게이트웨이가 처음 부팅될 때 필요한 설정 정보입니다. 이때 "endpoints" 필드는 외부와의 통신 정보이고, "applications" 필드는 게이트웨이에 등록된 애플리케이션의 정보입니다.

```

"staticResources": {
  "endpoints": [{
    "endpoint": {
      "physicalName": "poClientEndpoint",
      "protocol": "HTTP",
      "direction": "OUTBOUND",
      "bootState": "RUNNING",
      "connectType": "CLIENT",
      "httpUrl": {
        "split": true,
        "scheme": "http",
        "host": "${MS_IP}",
        "port": "${MS_PORT}"
      }
    }
  ]
},

```

```

{
  "endpoint": {
    "physicalName": "poServerEndpoint",
    "protocol": "HTTP",
    "bootState": "RUNNING",
    "connectionPool": {
      "max": "1000"
    },
    "httpUrl": {
      "split": true,
      "scheme": "http",
      "host": "${GW_IP}",
      "port": "${GW_PORT}"
    },
    "rulesetId": "anylink.system.SystemServiceRuleSet"
  }
},
{
  "endpoint": {
    "physicalName": "httpEndpoint",
    "protocol": "HTTP",
    "bootState": "Running",
    "connectType": "Server",
    "connection_pool": {
      "max": "100"
    },
    "httpUrl": {
      "scheme": "http",
      "port": "${HTTP_PORT}"
    },
    "rulesetId": "ApiApplication.ApiServiceGroup.ApiDefaultRuleChain",
    "errorRulesetId": "anylink.system.ApiGatewayErrorRuleSet"
  }
},
{
  "endpoint": {
    "physicalName": "httpsEndpoint",
    "protocol": "HTTP",
    "bootState": "Running",
    "connectType": "Server",
    "connectionPool": {
      "max": "100"
    },
    "useSsl": true,
    "ssl": {
      "keystore": {
        "storeType": "PEM",
        "storeLocation": "${SSL_STORE}"
      }
    },
    "httpUrl": {
      "scheme": "http",
      "port": "${HTTPS_PORT}"
    },
    "rulesetId": "ApiApplication.ApiServiceGroup.ApiDefaultRuleChain",
    "errorRulesetId": "anylink.system.ApiGatewayErrorRuleSet"
  }
}
}],
"applications": [{

```



```

    "id": <string>, (1)
    "version": <int>, (2)
    "servicegroups": [{ (3)
        "id": <string>, (4)
        "version": <int>, (5)
        "rulesets": [{ (6)
            "id": <string>, (7)
            "rule_chains": <RuleChainInfo> (8)
        }],
        "libpath": <string> (9)
    }]
}

```

각 설정 항목에 대한 설명은 다음과 같습니다.

항목	설명
(1) id	애플리케이션의 ID입니다.
(2) version	애플리케이션의 버전입니다.
(3) servicegroups	애플리케이션의 하위 서비스 그룹입니다.
(4) id	서비스 그룹의 ID입니다.
(5) version	서비스 그룹의 버전입니다.
(6) rulesets	서비스 그룹의 룰셋입니다.
(7) id	룰셋의 ID입니다.
(8) rule_chains	체인으로 정의한 룰셋의 실제 내용입니다.
(9) libpath	읽어들일 SharedLibrary의 경로입니다.



"endpoints" 필드의 설정 방법에 대한 자세한 내용은 [엔드포인트 설정](#)을 참고합니다.

• dynamicResources

게이트웨이가 웹 어드민으로부터 받아온 동적 설정 정보입니다. 해당 필드는 부팅 시 자동으로 설정되며, 웹 어드민을 통해서만 수정이 가능합니다.

```

"dynamicResources": {
    "string": <int>
    "endpoints": [<EndpointType>], (1)
    "applications": [<ApplicationType>], (2)
}

```

각 설정 항목에 대한 설명은 다음과 같습니다.

항목	설명
(1) endpoints	엔드포인트 목록입니다.

항목	설명
(2) applications	애플리케이션 목록입니다.

• EndpointInfo

게이트웨이에 등록된 endpoint 설정 정보입니다. 해당 필드는 웹 어드민을 통해서만 수정이 가능합니다.

```
"EndpointInfo": {
  "logical_name": <string>, (1)
  "physical_name": <string>, (2)
  "protocol": ("PO", "HTTP", "TCP"), (3)
  "direction": ("INBOUND", "OUTBOUND", "BOTH"),
  "boot_state": ("STOPPED", "RUNNING"),
  "http_url": { (4)
    "split": <bool>, (5)
    "url": <string>, (6)
    "scheme": ("HTTP", "HTTPS"), (7)
    "host": <string>, (8)
    "port": <string> (9)
  }
}
```

각 설정 항목에 대한 설명은 다음과 같습니다.

항목	설명
(1) logical_name	엔드포인트의 논리적인 이름입니다.
(2) physical_name	엔드포인트의 ID입니다.
(3) protocol	엔드포인트의 프로토콜입니다. (현재 HTTP만 사용 가능)
(4) http_url	연결할 원격 HTTP 서버의 URL입니다.
(5) split	URL의 분리 여부입니다. <ul style="list-style-type: none"> ◦ true: URL을 분리함 ◦ false: URL을 분리 안 함
(6) url	전체 URL입니다. 단, split이 'false'일 때 설정합니다.
(7) scheme	URL Scheme입니다. 단, split이 'true'일 때 설정합니다. <ul style="list-style-type: none"> ◦ HTTP ◦ HTTPS
(8) host	HTTP 호스트 이름입니다. 단, split이 'true'일 때 설정합니다.
(9) port	HTTP 포트 번호입니다. 단, split이 'true'일 때 설정합니다.

• EndpointGroupInfo

게이트웨이에 등록된 endpointgroup 설정 정보입니다. 해당 필드는 웹 어드민을 통해서만 수정이 가능합니다.

```
"EndpointGroupInfo":{
  "logical_name": <string>, (1)
  "physical_name": <string>, (2)
  "routing_type": ("ROUND_ROBIN", "WEIGHT_BASED", "PRIORITY"), (3)
  "routing": [{
    "endpoint_id": <string>, (4)
    "priority": <int>, (5)
    "weight": <int> (6)
  }]
}
```

각 설정 항목에 대한 설명은 다음과 같습니다.

항목	설명
(1) logical_name	엔드포인트의 논리적인 이름입니다.
(2) physical_name	엔드포인트의 ID입니다.
(3) routing_type	엔드포인트의 라우팅 정책입니다. (현재 WEIGHT_BASED만 지원)
(4) endpoint_id	라우팅을 설정할 하위 엔드포인트의 ID입니다.
(5) priority	엔드포인트의 우선순위입니다. 숫자가 낮을수록 우선됩니다.
(6) weight	엔드포인트의 가중치입니다. 숫자가 높을수록 많은 비율로 라우팅됩니다.

• loggerSystems

게이트웨이에서 사용하는 로거에 대한 설정 정보입니다. 해당 필드는 웹 어드민을 통해서만 수정이 가능합니다.

```
"loggerSystems": [{
  "path": "anylink",
  "level": "debug", (1)
  "rotatetype": "TIMEBASE", (2)
  "rotateparam": "daily", (3)
  "times": "local", (4)
  "format": "[%Y-%m-%d %L%H:%M:%S][%P][%q][%[THREAD]]: %t", (5)
  "expire": "none" (6)
}]
```

각 설정 항목에 대한 설명은 다음과 같습니다.

항목	설명
(1) level	<p>로거의 레벨입니다.</p> <ul style="list-style-type: none"> ◦ FATAL ◦ CRITICAL ◦ ERROR ◦ WARN ◦ NOTICE ◦ INFO ◦ DEBUG ◦ TRACE ◦ NONE
(2) rotatetype	<p>로거 로테이트 방식입니다.</p> <ul style="list-style-type: none"> ◦ SIZE ◦ TIMEBASE
(3) rotateparam	<p>로거 로테이트에 사용할 단위입니다.</p> <ul style="list-style-type: none"> ◦ never: 로테이트 안 함 ◦ <n>: 파일 크기가 <n> 바이트를 초과하면 로테이트 ◦ <n> K: 파일 크기가 <n> 킬로바이트를 초과하면 로테이트 ◦ <n> M: 파일 크기가 <n> 메가바이트를 초과하면 로테이트 ◦ [day,][hh:][mm]: 지정된 요일 및 시간에 로테이트 ◦ daily/weekly/monthly: 지정된 일간/주간/월간에 로테이트 ◦ <n> hours/weeks/months: 매 <n> 시간/주/월에 로테이트
(4) times	<p>로그 시간 타입입니다.</p> <ul style="list-style-type: none"> ◦ local: 현재 서버 시간 기준으로 로깅 ◦ UTC: UTC 시간 기준으로 로깅
(5) format	<p>로깅되는 포맷입니다.</p> <ul style="list-style-type: none"> ◦ [%Y-%m-%d %L%H:%M:%S]: 년-월-일 시:분:초 ◦ [%P]: 프로세스 ID ◦ [%q]: 로그 레벨 ◦ [%[THREAD]]: 스레드 정보 ◦ [%t]: 로그 내용

항목	설명
(6) expire	로그 파일의 만료기간입니다. <ul style="list-style-type: none"> ◦ none: 없음 ◦ [day,][hh:][mm] : 지정된 요일 및 시간 ◦ daily/weekly/monthly: 지정된 일간/주간/월간 ◦ <n> hours/weeks/months: 매 <n> 시간/주/월

• Monitoring

게이트웨이 모니터링에 대한 설정 정보입니다. 해당 필드는 웹 어드민을 통해서만 수정이 가능합니다.

```
"Monitoring": {
  "trace": {
    "enable" : "<bool>" (1)
    "option": {
      "fileEnable": "<bool>", (2)
      "fileName": "trace_log", (3)
      "path": "/var/log/anyapi", (4)
      "rotateParam": "30 M", (5)
      "rotateFileNum": "5", (6)
      "flushBufferSize": "32", (7)
      "interval": "3000",
      "endpoint": [ (8)
        "<grpc ip:grpc port>"
      ],
      "includeBody": "<bool>" (9)
    }
  },
  "stat": { (10)
    "enable": "<bool>",
    "option": {
      "endpoint": [
        "<grpc ip:grpc port>"
      ],
    }
  }
}
```

각 설정 항목에 대한 설명은 다음과 같습니다.

항목	설명
(1) enable	모니터링의 사용 여부입니다. <ul style="list-style-type: none"> ◦ true: 모니터링 ON ◦ false: 모니터링 OFF

항목	설명
(2) fileEnable	파일 저장 여부입니다. <ul style="list-style-type: none"> ◦ true: TRACE 정보를 파일로 저장 ◦ false: 파일 저장 사용 안 함
(3) fileName	저장될 파일 이름입니다.
(4) path	파일 저장 경로입니다.
(5) rotateParam	파일의 로테이션 크기입니다.
(6) rotateFileNum	로테이션되는 파일의 개수입니다.
(7) flushBufferSize	Span을 저장하는 버퍼 사이즈입니다.
(8) endpoint	gRPC의 IP 주소와 포트 번호입니다.
(9) includeBody	TRACE 정보에 body값의 포함 여부입니다.
(10) stat	통계 데이터의 전송 여부입니다. <ul style="list-style-type: none"> ◦ true: 전송 ◦ false: 전송 안 함

• loggerAccess

게이트웨이에서 사용하는 액세스 로거에 대한 설정 정보입니다. 해당 필드는 웹 어드민을 통해서만 수정이 가능합니다.

```
"loggerAccess": {
  "enable": "true", (1)
  "format": "%h %l %u %t %r %s %b", (2)
  "fileName": "access_log", (3)
  "path": "default", (4)
  "rotateType": "TIMEBASE", (5)
  "rotateParam": "daily", (6)
  "times": "local", (7)
  "expire": "none" (8)
}
```

각 설정 항목에 대한 설명은 다음과 같습니다.

항목	설명
(1) enable	액세스 로그의 사용 여부입니다. <ul style="list-style-type: none"> ◦ true: 사용 ◦ false: 사용 안 함

항목	설명
(2) format	<p>로깅되는 포맷입니다.</p> <ul style="list-style-type: none"> ◦ %h: 원격 호스트 주소 ◦ %l: 원격 로그인 이름 ◦ %u: Frank, HTTP 인증으로 알아낸 문서를 요청한 사용자의 ID ◦ %t: 서버 요청 처리를 마친 시간 ◦ %r: 요청의 첫 번째 줄 ◦ %s: 상태 ◦ %b: HTTP 헤더를 제외한 전송 바이트 수
(3) fileName	로그 파일의 이름입니다. (기본값: access_log)
(4) path	로그 파일이 저장되는 경로입니다. (기본값: \${GATEWAY_HOME}/protocol/HTTP/gw_id_test/logs/access_log)
(5) rotatetype	<p>로거 로테이트 방식입니다.</p> <ul style="list-style-type: none"> ◦ SIZE ◦ TIMEBASE
(6) rotateparam	<p>로거 로테이트에 사용할 단위입니다.</p> <ul style="list-style-type: none"> ◦ never: 로테이트 안 함 ◦ <n>: 파일 크기가 <n> 바이트를 초과하면 로테이트 ◦ <n> K: 파일 크기가 <n> 킬로바이트를 초과하면 로테이트 ◦ <n> M: 파일 크기가 <n> 메가바이트를 초과하면 로테이트 ◦ [day,][hh:][mm]: 지정된 요일 및 시간에 로테이트 ◦ daily/weekly/monthly: 지정된 일간/주간/월간에 로테이트 ◦ <n> hours/weeks/months: 매 <n> 시간/주/월에 로테이트
(7) times	<p>로그 시간 타입입니다.</p> <ul style="list-style-type: none"> ◦ local: 현재 서버 시간 기준으로 로깅 ◦ UTC: UTC 시간 기준으로 로깅
(8) expire	<p>로그 파일의 만료기간입니다.</p> <ul style="list-style-type: none"> ◦ none: 없음 ◦ [day,][hh:][mm] : 지정된 요일 및 시간 ◦ daily/weekly/monthly: 지정된 일간/주간/월간 ◦ <n> hours/weeks/months: 매 <n> 시간/주/월

• environment

게이트웨이의 타임아웃 관련 설정 정보입니다. 해당 필드는 웹 어드민을 통해서만 수정이 가능합니다.

```
"environment": {
  "connectionTimeout": "3000", (1)
  "readTimeout": "30000", (2)
  "failoverTimeout": "60000" (3)
}
```

각 설정 항목에 대한 설명은 다음과 같습니다.

항목	설명
(1) connectionTimeout	connectionTimeout 설정값입니다.
(2) readTimeout	readTimeout 설정값입니다.
(3) failoverTimeout	failoverTimeout 설정값입니다.

1.3.2. 디렉터리 구조

다음은 게이트웨이를 설치하고, 한번 실행한 후에 사용하는 디렉터리 구조입니다.

```
${GATEWAY_HOME}
|-- bin
|-- inc
|-- lib
|-- config
|-- protocol
|   |-- ${PROTOCOL}
|   |   |-- ${GATEWAY_NAME}
|   |-- logs
|-- uds
```

bin

AnyAPI 게이트웨이의 부팅을 위한 바이너리가 위치합니다.

inc

AnyAPI 게이트웨이 헤더 파일들이 위치합니다.

lib

AnyAPI 게이트웨이에서 사용하는 라이브러리들이 위치합니다.

config

AnyAPI 게이트웨이 설정 파일이 위치합니다.

protocol

AnyAPI 게이트웨이에서 사용하는 리소스가 위치합니다.

다음은 하위 경로에 대한 설명입니다.

하위 경로	설명
\${PROTOCOL}	게이트웨이가 사용하는 프로토콜 이름이 표시됩니다. AnyAPI에는 HTTP로 고정됩니다.
\${GATEWAY_NAME}	게이트웨이 리소스가 저장되는 디렉터리입니다.
logs	lib 디렉터리의 export 대상 디렉터리입니다.

uds

도메인 소켓 통신을 하는 경우 필요한 파일이 위치합니다.

1.3.3. 환경 변수

다음은 게이트웨이를 구동하기 위해 필요한 환경 변수에 대한 설명입니다.

환경변수	설명
GATEWAY_HOME	게이트웨이가 설치된 홈 디렉터리입니다. (필수 사항) <div>GATEWAY_HOME=/home/anylink/gateway</div>
GATEWAY_ENCODING	게이트웨이 인코딩 방식입니다. <div>GATEWAY_ENCODING=utf8</div>
GATEWAY_INACTIVE_CACHE	게이트웨이의 캐싱 여부입니다. <ul style="list-style-type: none"> ◦ true: 캐싱 비활성화 ◦ false: 캐싱 활성화 <div>GATEWAY_INACTIVE_CACHE=false</div>
LD_LIBRARY_PATH	lib 디렉터리를 export할 대상 경로입니다. <div>export LD_LIBRARY_PATH=\${LD_LIBRARY_PATH}:\${GATEWAY_HOME}/lib</div>
CURLPOT_LOW_SPEED_TIME	CURLOPT_LOW_SPEED_LIMIT 옵션으로 지정된 속도 이하로 전송될 수 있는 최대 시간입니다. 이때 초 단위로 지정하며, 해당 시간 이상 지속되면 전송이 취소됩니다.
CURLOPT_LOW_SPEED_LIMIT	최하 전송 속도입니다. 이때 초당 전송될 바이트 단위로 지정하며, 해당 속도가 CURLOPT_LOW_SPEED_TIME 옵션으로 지정된 시간만큼 지속되면 전송이 취소됩니다.

환경변수	설명
GATEWAY_BINDING_RETRYTIME	<p>바인딩 실패 후 다시 바인딩을 시도하기 전까지 대기하는 시간입니다.</p> <pre>export GATEWAY_BINDING_RETRYTIME=160000(ms)</pre>
GATEWAY_BINDING_RETRYCOUNT	<p>다시 바인딩을 시도하는 횟수입니다.</p> <p>모든 횟수를 실패하면 프로세스가 강제 종료됩니다.</p> <ul style="list-style-type: none"> ◦ 0: 재시도 없이 프로세스 종료 ◦ -1: 무한 바인딩 시도 <pre>export GATEWAY_BINDING_RETRYCOUNT=3</pre>

2. 엔드포인트 구성

본 장에서는 AnyAPI 게이트웨이의 엔드포인트 구성과 각 엔드포인트의 구성 요소들의 배포 방법에 대해 설명합니다.

2.1. 개요

엔드포인트는 실제로 외부 시스템과 연결되어 데이터를 송수신하는 기능을 담당합니다.

보통 하나의 엔드포인트는 하나의 다른 시스템과 연결할 수 있으며 하나의 게이트웨이에 엔드포인트를 여러 개 두어 여러 시스템과 연결할 수 있습니다. 엔드포인트 리소스는 엔드포인트 그룹과 엔드포인트, 두 가지의 리소스로 분리하여 관리합니다. 엔드포인트는 AnyLink 게이트웨이에서 나온 개념이지만 AnyAPI에서는 해당 개념을 활용하여 AnyAPI 게이트웨이로써 동작하고 있습니다.

2.1.1. 주요 특징

엔드포인트에서는 외부에 대한 정보(IP 주소, 포트 번호, 타임아웃과 같은 운영 정보)를 관리합니다. 엔드포인트는 프로토콜마다 설정 포인트가 조금씩 다르나 AnyAPI에서는 PO와 HTTP 프로토콜만을 다룹니다.

2.1.2. AnyAPI 기본 엔드포인트

AnyAPI 게이트웨이에서는 기본적으로 4개의 엔드포인트가 사전에 staticResources 내에 정의되어 있으며 웹 어드민을 통해 서버를 추가할 때마다 dynamicResources에 엔드포인트들이 추가됩니다.

다음은 AnyAPI 게이트웨이에서 기본적으로 설정되어 있는 4가지 엔드포인트에 대한 설명입니다.

- **poClientEndpoint**

poClientEndpoint는 Master의 애플리케이션인 Admin과 통신하기 위해 만들어진 클라이언트 엔드포인트다. 게이트웨이는 부팅할 때 해당 엔드포인트를 통하여 Admin과 연결을 맺고 GatewayPost 서비스를 요청하여 연결을 수립합니다. 연결이 수립되면 Admin은 들고 있는 전체 설정 정보를 배포하게 되며 이후 설정이 변경될 때마다 부분 배포가 이루어집니다. 또한 실제 API가 동작하게 될 때 이 연결을 통해서 Master에 인증/인가 등을 요청합니다.

- **poServerEndpoint**

poServerEndpoint는 Master가 Admin에서 관리하는 라우팅 정보를 배포 받기 위해 사용하는 서버 엔드포인트다. 해당 엔드포인트를 통해서 설정 파일을 배포하거나 받을 수 있습니다.

- **httpEndpoint / httpsEndpoint**

httpEndpoint와 httpsEndpoint는 실제 클라이언트의 요청을 받는 HTTP 포트에 대한 설정입니다. httpEndpoint는 HTTP, httpsEndpoint는 HTTPS에 대한 포트입니다.

2.2. 엔드포인트 설정

엔드포인트는 게이트웨이의 설정 파일 내에 복수의 엔드포인트를 설정할 수 있습니다. (현재 JSON 형식만 지원)

다음은 엔드포인트의 설정 예시 및 설정 항목에 대한 설명입니다.

• PO 엔드포인트 설정

```
"endpoint": {
  "physicalName": "poClientEndpoint",
  "protocol": "HTTP",
  "direction": "OUTBOUND",
  "bootState": "RUNNING",
  "connectType": "CLIENT",
  "httpUrl": {
    "split": true,
    "scheme": "http",
    "host": "${MS_IP}", (1)
    "port": "${MS_PORT}" (2)
  }
}
```

항목	설명
(1) host	연결할 외부 주소입니다. 단, 클라이언트일 때 설정합니다. (보통 JEUS의 MS IP를 사용)
(2) port	연결에 사용할 포트 번호입니다. (보통 JEUS의 MS HTTP-PORT를 사용) 게이트웨이의 서버, 클라이언트 모드에 따라 의미가 다릅니다. <ul style="list-style-type: none">◦ SERVER: 연결할 외부 포트◦ CLIENT: 게이트웨이에서 열 내부 포트

• SSL 엔드포인트 설정

```
"endpoint": {
  "physicalName": "sslServer",
  "protocol": "HTTP",
  "direction": "Inbound",
  "bootState": "Running",
  "connectType": "Server",
  "idleTimeout": "360000", (1)
}
```

항목	설명
(1) idleTimeout	소켓에 응답이 없을 경우 세션이 종료되는 시간(ms)을 설정합니다. (기본값: 360000)

• HTTPS 엔드포인트 설정

```
"endpoint": {
```

```

    "physicalName": "httpsEndpoint",
    "protocol": "HTTP",
    "direction": "Inbound",
    "bootState": "Running",
    "connectType": "Server",
    "connection_pool": {
        "max": "100"
    },
    "useSsl": true, (1)
    "ssl": { (2)
        "keystore": { (3)
            "storeType": "PEM", (4)
            "storeLocation": "${SSL_STORE}" (5)
        }
    },
    "httpUrl": {
        "scheme": "https",
        "port": "${HTTPS_PORT}" (6)
    },
    "rulesetId": "ApiApplication.ApiServiceGroup.ApiDefaultRuleChain",
    "errorRulesetId": "anylink.system.ApiGatewayErrorRuleSet"
}

```

항목	설명
(1) useSsl	엔드포인트의 SSL 적용 여부입니다.
(2) ssl	해당 엔드포인트에서 사용할 SSL에 대한 세부 정보입니다.
(3) keystore	SSL 관련 Key Store 정보입니다. 해당 필드는 SSL 인증에 사용될 Key Store 파일 설정과 연관됩니다.
(4) storeType	사용할 인증서 타입입니다. (현재 PEM 타입만 가능)
(5) storeLocation	인증서가 저장된 파일 경로입니다.
(6) port	엔드포인트에서 열 내부 HTTPS 포트입니다.

• 공통 엔드포인트 설정 (PO, HTTP)

항목	설명
physicalName	엔드포인트의 물리적인 이름입니다.
protocol	엔드포인트에서 사용하는 프로토콜입니다.
direction	엔드포인트의 방향입니다. <ul style="list-style-type: none"> ◦ OUTBOUND: 외부로 나가는 엔드포인트 ◦ INBOUND: 외부에서 들어오는 엔드포인트 ◦ BOTH: 외부와 양방향으로 통신이 가능한 엔드포인트
bootState	게이트웨이 부팅 시 엔드포인트의 초기 상태입니다. <ul style="list-style-type: none"> ◦ Running: 부팅할 때 자동으로 시작합니다. ◦ Stopped: 부팅할 때 시작하지 않습니다.

항목	설명
connectType	<p>게이트웨이의 연결 타입입니다.</p> <ul style="list-style-type: none"> ◦ CLIENT: 클라이언트 타입의 엔드포인트 ◦ SERVER: 서버 타입의 엔드포인트
connection_pool	<p>게이트웨이의 연결 풀입니다. 연결 풀은 엔드포인트가 여러 개의 물리적인 통신 연결을 가질 수 있기 때문에 이러한 물리적인 연결을 어떻게 관리할지 설정할 수 있습니다.</p> <ul style="list-style-type: none"> ◦ min: 연결 풀이 최소로 들고 있어야 하는 연결의 개수를 양의 정수로 지정합니다. 항상 이 연결 수 만큼 유지해야하므로 부팅이 된 엔드포인트는 이 값의 연결이 생길 때까지 연결을 시도합니다. 단, 클라이언트 타입일 때만 동작합니다. ◦ max: 연결 풀이 최대로 유지할 수 있는 연결 개수를 양의 정수로 지정합니다.
rulesetId	해당 엔드포인트로 요청이 들어왔을 때 실행시킬 룰입니다.
errorRulesetId	해당 엔드포인트로 요청을 처리하는 도중 에러가 발생하였을 때 처리할 룰입니다.

부록 A: 문제 해결

본 부록에서는 AnyAPI 게이트웨이 관련 오류에 대한 원인과 해결 방법에 대해 설명합니다.

A.1. FailedGatewayReadsizeBelowZero

원인	SSL 클라이언트쪽에서 연결을 해제 시 발생합니다.
조치사항	openssl s_client 명령을 사용하여 SSL 인증서가 올바른지 확인합니다. <div>openssl s_client -connect IP:port</div>

A.2. GatewayAPIAuthenticationFailException

원인	단말 인증에 실패 또는 토큰 값이 다르거나 토큰의 유효기한이 지났을 때 발생합니다.
조치사항	토큰이 정확한지 확인합니다.

A.3. GatewayAPINoMatchingConditionException

원인	클라이언트가 올바른 경로로 호출하지 않았을 때 발생합니다.
조치사항	올바른 경로와 메소드로 호출하였는지 확인합니다. 만약 올바른 경로로 호출하였다면 Admin으로부터 정상 배포가 되었는지 확인하고, DEBUG 로그의 "[ParsingRule] regex match" 로그를 확인하여 제대로 비교되는지 확인합니다.

A.4. GatewayEndpointNotExistException

원인	라우팅할 백엔드 서버가 존재하지 않을 때 발생합니다.
조치사항	백엔드 서버가 제대로 동작하는지 확인하고, 로그 상에서 failover가 발생했는지 확인합니다.

A.5. GatewaySslInitFailException

원인	SSL 인증서의 초기화를 실패했을 때 발생합니다.
조치사항	인증서의 경로 및 비밀번호가 맞는지 확인합니다.

A.6. SSL unknown error

원인	SSL 서버(게이트웨이)에서 발생한 에러로, 잘못된 SSL 요청을 할 경우 발생합니다. (예: https 포트에 http 요청)
조치사항	https 요청을 올바르게 했는지 클라이언트 요청을 확인합니다.

A.7. Invalid HTTP method

원인	HTTP를 파싱할 때 발생하는 에러입니다.
조치사항	HTTP 요청 메시지가 정확한지 확인합니다. error message는 실제 에러가 발생한 메시지를 출력하며 error location은 파싱 문제가 발생한 위치를 말합니다.