관리자 안내서

WebtoB 6

TMAXSOFT

저작권 공지

제한된 권리

상표 공지

오픈 소스 소프트웨어 공지

기술 지원

홈페이지: https://www.tmaxsoft.com

기술 서비스 센터: 1544-8629

안내서 이력

제품 버전	안내서 버전	발행일	비고
WebtoB 6	3.1.1	2024-10-07	-

목차

1. 시작하기

1.1. 라이선스별 지원 기능

WebtoB는 라이선스 타입별로 다음과 같은 기능을 제공합니다.

라이선스 타입	설명
TRIAL	인스톨러로 설치한 WebtoB에는 기본적으로 트라이얼 라이선스(Trial License)가 내장되어 있습니다. HTH는 1개, 최대 사용자 수는 5개로 제한됩니다.
BASE	JEUS의 내장 WebtoB로 사용되는 경우입니다. HTH가 1개로 제한됩니다. UNIX/Linux 환경에서는 도메인 소켓(named pipe)을 통해서만 JEUS와 연동할 수 있기 때문에 JSVPort를 사용하지 않습니다.
STANDARD	BASE와 달리 HTH가 1개 이상 지원되는 일반적인 경우입니다. JEUS와도 자유롭게 연동할 수 있으며 WebtoB가 제공하는 대부분의 기능을 사용할 수 있습니다.
ENTERPRISE	STANDARD에서 제공하는 모든 기능 이외에 아래와 같은 추가적인 기능을 사용할 수 있습니다. ◦ 내장된 JEUS의 JSP/Servlet Engine을 사용할 수 있습니다. 이 경우 내장 JEUS의 1개 서버(DAS)를 사용할 수 있습니다. ◦ 역방향 프록시를 이용한 다른 WAS를 연동할 때 다중 구성을 위한 역방향 프록시 그룹 기능을 사용할 수 있습니다. ◦ WebDAV를 위한 HTTP 메소드(MKCOL, COPY, MOVE, PORPFIND)를 사용할 수 있습니다. ◦ 필터(Filter) 처리를 위한 FILTERS 프로세스를 사용할 수 있습니다. 특히 CA사의 SSO 연동을 위한 SiteMinder Filter(wbSmISAPI)를 사용할 수 있습니다. ◦ WebAdmin을 사용할 수 있습니다.

1.2. 빠른 시작

WebtoB 설치 후 신속하게 정상 동작을 확인하려면 다음과 같은 간단한 절차를 따라 진행할 수 있습니다.

1. WebtoB의 설정 파일 검증 툴인 **configValidator**을 사용해서 설정 파일이 올바르게 구성되었는지 확인합니다.

```
$ configValidator
```

다음은 WEBTOB6_HOME_PATH\config\webtob-config.json 파일의 설정 예시입니다.

```
{
  "node": {
    "name": "webtob_node",
    "hth_count": 1,
    "worker_threads": 8,
    "hth_schedule": "RR"
  },
  "server": {
    "http": {
      "common_config": {
        "doc_root": "docs",
        "service_order": "uri,ext",
        "access_log": "access_log1"
      },
      "http_servers": [
        {
          "name": "http1",
          "port": 8080,
          "enable_ssl": true,
          "ssl_name": [
            "ssl1"
          ]
        }
      ]
    },
    "wjp": {
      "port": 9900,
      "wjp_servers": [
        {
          "name": "MyGroup1",
          "svr_chk_time": 60
        }
      ]
    }
  },
  "logging": {
    "access_log": [
      {
        "name": "access_log1",
        "level": "INFO",
        "format": "DEFAULT",
        "handlers": {
          "file_handler": {
            "file_name": "logs/access.log"
          },
          "enable_console_handler": false
        }
      }
    ],
    "error_log": {
      "level": "INFO",
      "handlers": {
        "file_handler": {
          "file_name": "logs/errors.log"
        },
        "enable_console_handler": false
```

```
      }
    },
    "system_log": [
      {
        "name": "webtob",
        "level": "INFO",
        "handlers": {
          "file_handler": {
            "file_name": "logs/webtob.log"
          },
          "enable_console_handler": true
        }
      }
    ]
  },
  "destination": {
    "jeus": [
      {
        "name": "MyGroup1",
        "server_schedule": "RR",
        "connection_schedule": "RR"
      }
    ],
    "reverse_proxy": {
      "reverse_proxy_group": [
      {
        "name": "rproxyGroup1",
        "reverse_proxy_server": [
          {
            "address": "internal.server:80"
          }
        ]
      }
    ]},
    "htmls": [
      {
        "name": "htmls1"
      }
    ]
  },
  "service": {
    "uri": [
      {
        "name": "uri1",
        "target_http_servers": [
          "http1"
        ],
        "match": {
          "type": "prefix",
          "target": "/rproxy",
          "rewrite": "/"
        },
        "destination": {
          "type": "REVERSE_PROXY",
          "target": "rproxyGroup1"
        }
      }
    ],
    "ext": [
```

```
      {
        "name": "ext1",
        "target_http_servers": [
          "*"
        ],
        "match": {
          "type": "exact",
          "target": "text/html"
        },
        "destination": {
          "type": "JEUS",
          "target": "MyGroup1"
        }
      }
    ]
  },
  "ssl": {
    "ssl_configs": [
      {
        "name": "ssl1",
        "webtob_certificate_file": "server.crt",
        "webtob_certificate_key_file": "key.crt"
      }
    ]
  }
}
```

2. 명령 프롬프트에 WebtoB를 기동 명령을 입력합니다.

```
$ wsboot
```

3. 브라우저를 열고, 아래와 같이 URL을 입력합니다.

```
http://<IP Address>:<8080 또는 사용자 지정 포트 번호>/
```

정상적으로 WebtoB가 기동된 경우 아래와 같이 화면이 표시됩니다.

4. 명령 프롬프트에 WebtoB를 종료 명령을 입력합니다.

```
$ wsdown
```

2. 환경 설정

2.1. 기본 구조

WebtoB 환경 설정은 하나의 WebtoB 노드(Node)와 웹 서버의 동작에 대해 설정합니다.

WebtoB의 환경 설정 파일은 YAML과 JSON 포맷을 지원합니다. 이때 "절 이름", "항목 이름", "설정값"으로 구성됩니다.

다음은 JSON 포맷의 설정 형태입니다.

```
"절 이름": {
  "항목 이름" : 설정값,
  ...,
  "항목 이름" : 설정값
}
```

절을 시작한 이후 정의할 이름을 설정하고, 각 절에서 설정할 수 있는 각각의 항목을 설정합니다.

환경 설정은 포맷에 따라 JSON 또는 YAML 작성 규칙에 준수해서 설정합니다.

다음은 JSON 포맷의 설정 예시입니다.

```
{
  "node": {
    "name": "webtob_node",
    "hth_count": 1,
    "worker_threads": 8
  },
  "server": {
    "http": {
      "common_config": {
        "doc_root": "docs",
        "access_log": "access_log1"
      },
      "http_servers": [
        {
          "name": "http1",
          "port": 8080,
          "enable_ssl": true,
          "ssl_name": [
            "ssl1"
          ]
        }
      ]
    },
    "wjp": {
      "port": 9900,
```

```
      "wjp_servers": [
        {
          "name": "MyGroup1"
        }
      ]
    }
  },
  "logging": {
    "access_log": [
      {
        "name": "access_log1",
        "level": "INFO",
        "format": "DEFAULT",
        "handlers": {
          "file_handler": {
            "file_name": "logs/access.log"
          }
        }
      }
    ],
    "error_log": {
      "level": "INFO",
      "handlers": {
        "file_handler": {
          "file_name": "logs/errors.log"
        }
      }
    },
    "system_log": [
      {
        "name": "webtob",
        "level": "INFO",
        "handlers": {
          "file_handler": {
            "file_name": "logs/webtob.log"
        }
      }
    ]
  },
  "destination": {
    "jeus": [
      {
        "name": "MyGroup1"
      }
    ],
    "reverse_proxy": {
      "reverse_proxy_group": [
      {
        "name": "rproxyg1",
        "reverse_proxy_server": [
          {
            "address": "internal.server:80",
            "name": "rproxy1"
          }
        ]
      }
    ]},
    "htmls": [
      {
```

```
        "name": "htmls1"
      }
    ]
  },
  "service": {
    "uri": [
      {
        "name": "uri1",
        "target_http_servers": [
          "http1"
        ],
        "match": {
          "type": "prefix",
          "target": "/rproxy",
          "rewrite": "/"
        },
        "destination": {
          "type": "REVERSE_PROXY",
          "target": "rproxyGroup1"
        }
      }
    ],
    "ext": [
      {
        "name": "ext1",
        "target_http_servers": [
          "*"
        ],
        "match": {
          "type": "exact",
          "target": "text/html"
        },
        "destination": {
          "type": "JEUS",
          "target": "MyGroup1"
        }
      }
    ]
  },
  "ssl": {
    "ssl_configs": [
      {
        "name": "ssl1",
        "certificate_file": "server.crt",
        "certificate_key_file": "key.crt"
      }
    ]
  }
}
```

2.1.1. 설정 가능한 절의 종류

다음은 WebtoB에서 설정해야 하는 절에 대한 설명입니다.

절	설명
ACCESS 절	클라이언트가 접속을 시도할 때 IP 주소, 네트워크/넷마스크, 헤더 정보를 기준으로 요청의 허용 여부를 설정합니다.
ALIAS 절	실제 서버의 물리적 디렉터리 경로와 URI를 연결하도록 설정합니다.
DESTINATION 절	클라이언트 요청을 실제로 처리할 내부 서버에 대해 설정합니다.
ERRORDOCUMENT 절	에러 문제가 발생했을 때 대응 방법을 설정합니다.
FILTER 절	Filter 모듈을 사용할 FILTER 절을 설정합니다.
HEADERS 절	HTTP 요청 및 응답의 헤더를 제어할 때 설정합니다.
LOGGING 절	클라이언트의 요구 내역을 기록하는 형식을 지정합니다.
NODE 절	WebtoB 노드에 대한 구체적인 환경에 대해 설정합니다.
SERVER 절	클라이언트가 접속할 수 있는 정보에 대해 설정합니다.
SERVICE 절	HTTP 요청을 처리할 내부 서버를 요청의 URI, EXT 기반으로 지정합니다.
SSL 절	WebtoB에서 사용할 SSL의 기능을 설정합니다.

DESTINATION, SERVER, SERVICE 절은 반드시 설정해야 합니다.

2.1.2. 설정 항목 값의 형식 및 설정 방법

항목의 설정값은 항목에 따라 object, integer, string, boolean, array 형식으로 설정하며, 각각 다음과 같이 설정합니다.

종류	설명
object	하위에 여러 설정을 갖는 경우 설정합니다. ◦ JSON 포맷: 중괄호({ })로 묶기 ◦ YAML 포맷: 개행 후 한 단계 들여쓰기
integer	숫자를 설정합니다.
string	문자열을 설정합니다.
boolean	true 또는 false로 설정합니다.
array	동일한 형식의 설정을 여러 개 설정합니다. 대괄호([ ])로 묶어서 각 설정을 콤마(,)로 구분하거나, 개행 후 하이픈(-)으로 구분하여 표현할 수 있습니다.

본 안내서에서 설명하는 설정 항목의 내용을 제대로 이해하려면, 아래의 기본 규칙을 먼저 숙지해야 합니다.

- 이름 앞에 해시 기호(#)로 시작하는 항목은 선택 항목입니다.
- 형식이 대괄호([ ])로 쌓여 있으면 특정 형식으로 구성된 array 형식을 의미합니다.

- 기본값이 있는 설정의 경우 해시 기호(#) 이후에 표시합니다.
- integer 형식의 경우 괄호를 사용하여 설정 가능한 범위를 표시합니다.
- 최댓값 제한이 없을 경우 생략하며, INT_MAX(2147483647)까지만 사용할 수 있습니다.
- "$ENV"는 환경 변수를 참조할 수 있다는 의미입니다.
- "R.PATH"는 상대 경로로 설정하는 경우 $WEBTOB6_HOME_PATH로부터의 상대 경로로 사용한다는 의미입니다.
- object 형식의 경우 중괄호 ({ })로 내부 설정에 대해 표현합니다.
- object 형식의 경우 같은 형식이 상위 설정에 있을 경우 "{...}" 로 설명을 생략합니다.
- "COMMON"는 설정하지 않은 경우 상위 설정을 따른다는 의미입니다.

2.2. ACCESS 절

클라이언트에서 접속을 시도할 때 IP 주소, 네트워크/넷마스크, 헤더 정보들을 기준으로 요청의 허용/제한을 설정합니다. 또한 요청 허용/제한이 적용되는 순서를 설정할 수 있습니다.

ACCESS 절은 SERVICE 절, SERVER 절에 적용될 수 있으며, 각각에서 정의한 리소스를 허용/제한합니다.

2.2.1. 설정 항목

다음은 ACCESS 절의 환경 설정 형식입니다.

```
#"access": {
    "access_list": [
        {
            "name": string,
            "policy": string,
            #"allow_network": [string],
            #"deny_network": [string],
            #"allow_header": [string],
            #"deny_header": [string],
            #"method_whitelist": [string],
            #"method_blacklist": [string]
        }
    ]
}
```

절과 설정 항목의 구성에 대한 기호나 내용에 대한 자세한 내용은 설정 항목 값의 형식 및 설정 방법을 참고합니다.

access_list

ACCESS 설정에 대한 목록입니다.

구분	설명
자료형	array(object)

access_list/name (필수 항목)

ACCESS 절 이름을 설정합니다.

구분	설명
자료형	string
범위	255자 이내

access_list/policy (필수 항목)

method, allow_network, allow_header, deny_network, deny_header가 적용되는 순서를 설정합니다.

구분	설명
자료형	string
범위	"blacklist" | "whitelist" | "method_only"

access_list/allow_network

요청이 허용되는 IP 주소나 네트워크/넷마스크를 설정합니다. 이때 Allow = "all"은 특수 값으로서 모든 IP 주소를 의미합니다.

구분	설명
자료형	array(string)
범위	256개 이내(255자 이내)

access_list/deny_network

요청이 거절되는 IP 주소나 네트워크/넷마스크를 설정합니다.

구분	설명
자료형	array(string)
범위	256개 이내(255자 이내)

access_list/allow_header

요청이 허용되는 헤더를 설정합니다. 이때 요청에 포함된 <header field name> 헤더 값이 <regular expression>의 패턴과 매치됩니다.

구분	설명
자료형	array(string)
범위	256개 이내(255자 이내)
설정 형식	<header field name> <regular expression>

access_list/deny_header

요청이 거절되는 헤더를 설정합니다. 이때 요청에 포함된 <header field name> 헤더 값이 <regular expression>의 패턴과 매치됩니다.

구분	설명
자료형	array(string)
범위	256개 이내(255자 이내)
설정 형식	<header field name> <regular expression>

access_list/method_whitelist

적용할 HTTP 메소드를 설정합니다. 단, method_blacklist와 동시에 설정할 수 없습니다.

구분	설명
자료형	array(string)
범위	"GET", "POST", "PUT", "HEAD", "DELETE", "CONNECT", "OPTIONS", "TRACE", "PATCH", "PROPFIND", "PROPPATCH", "MKCOL", "COPY", "MOVE", "LOCK", "UNLOCK"

access_list/method_blacklist

제외할 HTTP 메소드를 설정합니다. 단, method_whitelist와 동시에 설정할 수 없습니다.

구분	설명
자료형	array(string)
범위	"GET", "POST", "PUT", "HEAD", "DELETE", "CONNECT", "OPTIONS", "TRACE", "PATCH", "PROPFIND", "PROPPATCH", "MKCOL", "COPY", "MOVE", "LOCK", "UNLOCK"

2.2.2. 설정 예시

다음은 ACCESS 절을 설정한 예시입니다.

```
{
    "access": {
```

```
        "access_list": [
            {
                "name": "access1",
                "policy": "blacklist",
                "deny_network": [ "192.168.1.43/255.255.255.0" ]
            }
        ]
    }
}
```

2.3. ALIAS 절

실제 서버의 물리적 디렉터리 경로와 URI를 연결하도록 설정할 수 있습니다. 즉, 어떤 특정한 URI에 대한 요구가 들어오면 이를 실제의 물리적인 디렉터리에 매핑시켜서 이곳에서 원하는 리소스를 찾아 처리하게 하는 방식입니다. 이는 사용자가 Document root에 상관없이 지정할 수 있기 때문에 관리하는 입장에서 편리한 기능입니다.

2.3.1. 설정 항목

다음은 ALIAS 절의 환경 설정 형식입니다.

```
#"alias": {
    #"alias_list": [
        {
            "name": string,
            "url": string,
            "real_path": string
        }
    ]
}
```

절과 설정 항목의 구성에 대한 기호나 내용에 대한 자세한 내용은 설정 항목 값의 형식 및 설정 방법을 참고합니다.

alias_list

ALIAS 설정에 대한 목록입니다.

구분	설명
자료형	array(object)

alias_list/name (필수 항목)

ALIAS 설정의 이름입니다. 다른 절에서 ALIAS 절의 기능을 사용할 때는 이 'name'을 설정해야 합니다.

구분	설명
자료형	string
범위	31자 이내

alias_list/url (필수 항목)

Alias로 지정할 URI를 설정합니다. 요청 URL의 첫 번째 byte부터 설정값의 URL 길이만큼 비교하여 매칭 여부를 확인합니다.

구분	설명
자료형	string
범위	255자 이내
설정 형식	/로 시작하고 /로 끝나야 합니다.

alias_list/real_path (필수 항목)

서버 안의 물리적 디렉터리의 경로명을 설정합니다. 이때 상대 경로(/로 시작하지 않는 경로)는 자동으로 "$WEBTOB6_HOME_PATH/상대 경로"로 대체됩니다.

구분	설명
자료형	string
범위	255자 이내
설정 형식	/로 끝나야 합니다.

2.3.2. 설정 예시

다음은 ALIAS 절을 설정한 예시입니다.

```
{
    "alias": {
        "alias_list": [
            {
                "name" : "alias1",
                "url": "/external/path/",
                "real_path": "/internal/real/path/"
            }
        ]
    }
}
```

2.4. DESTINATION 절

요청을 전달하고 응답을 받는 서비스 수행 시 사용될 백엔드 서버를 설정합니다. 이때 **'jeus'**, **'reverse proxy'**, **'htmls'** 중 하나는 반드시 있어야 합니다.

2.4.1. 설정 항목

다음은 DESTINATION 절의 환경 설정 형식입니다.

```
"destination": {
    #"jeus": [
        {
            "name": string,
            #"server_schedule": string,                                  # "RR"
            #"connection_schedule": string,                              # "RR"
            #"enable_flexible_sticky_session_routing": boolean,         # false
            #"enable_request_level_ping": boolean,                      # false
            #"request_level_ping_timeout": integer                      # 3 (0-INT_MAX)
            #"request_level_ping_retry_count": integer                  # 0 (0-INT_MAX)
            #"headers": [string],
            #"session_id_cookie_key": string,                           # "JSESSIONID"
            #"rewrite_cookie_domain": string,
            #"rewrite_cookie_path": {
                "from": string,
                "to": string
            },
            #"enable_cache": boolean,                                   # false
            #"cache_refresh": integer,                                  # 3600 (1-INT_MAX)
            #"max_queue_count": integer,                                # 0 (0-INT_MAX)
            #"max_queue_url": string,
            #"max_queue_status_code": integer,                          # 503
            #"queue_timeout": integer,                                  # 0 (0-INT_MAX)
            #"backup_server": string
        }
    ],
    #"reverse_proxy": {
        #"common_config": {                                             # COMMON
            #"headers": [string],
            #"compression": [string],
            #"compression_min_size": integer,                           # 0 (0-INT_MAX)
            #"rewrite_redirect": [
                {
                    "original_uri": string,
                    "redirect_path": string
                }
            ],
            #"rewrite_cookie_domain": string,
            #"rewrite_cookie_path": {
                "from": string,
                "to": string,
            },
            #"max_queue_count": integer,                                # 0 (0-INT_MAX)
            #"max_queue_url": string,
            #"max_queue_status_code": integer,                          # 503
            #"websocket_connections_max": integer,                      # 0 (0-INT_MAX)
```

```
        #"websocket_session_timeout": integer,                      # 0 (0-INT_MAX)
        #"set_host_header": string,
        #"queue_timeout": integer,                                  # 0 (0-INT_MAX)
        #"name_resolution_interval": integer,                       # 0 (0-INT_MAX)
        #"server_health_check": {
            #"retry_count": integer,                                # 3 (0-INT_MAX)
            #"failback_interval": integer,                          # 60 (0-INT_MAX)
            #"failure": [string],                                   # ["connection_timeout"]
            #"enable_name_resolution_on_fail": boolean              # true
        },
        #"rewrite_html_url": [
            {
                "tag": string,
                "attribute": [string],
                "from": string,
                "to": string
            }
        ],
        #"rewrite_html_max_size": integer                           # 10240 (1-INT_MAX)
    },
    "reverse_proxy_group": [
        {
            "name": string,
            "reverse_proxy_server": [
                {
                    "name": string,
                    "address": string,
                    #"enable_proxy_ssl": boolean,                   # false
                    #"proxy_ssl_name": [string],
                    #"persistent_server_connections_min": integer,  # 0 (0-INT_MAX)
                    #"persistent_server_connections_max": integer,  # 0 (0-INT_MAX)
                    #"persistent_server_check_time": integer,       # 30 (0-INT_MAX)
                    #"persistent_server_check_url": string,
                    #"persistent_server_timeout": integer,          # 300 (0-INT_MAX)
                    #"enable_cache": boolean,                       # false
                    #"cache_refresh": integer,                      # 3600 (1-INT_MAX)
                    #"sticky_session_routing_id": string,
                    #"is_backup_server": boolean,                   # false
                    #"load_balancing_factor": integer,              # 1 (1-INT_MAX)
                    #"common_config": {...}                         # COMMON
                }
            ],
            #"server_schedule": string,                             # "RR"
            #"sticky_session_routing": {
                #"policy": string,                                  # "UseOriginalCookie"
                #"session_id_cookie_key": string,                   # "JSESSIONID"
                #"enable_flexible_sticky_session_routing": boolean  # false
            },
            #"common_config": {...}                                 # COMMON
        }
    ]
},
#"htmls":[
    {
        "name": string,
        #"headers": [string],
        #"compression": [string],
        #"compression_min_size": integer,                           # 0 (0-INT_MAX)
        #"enable_cache": boolean,                                   # false
```

```
            #"cache_refresh": integer,                                  # 3600 (1-INT_MAX)
            #"queue_timeout": integer,                                  # 0 (0-INT_MAX)
            #"enable_sendfile": boolean,                                # false
            #"sendfile_min_size": integer,                              # 0 (0-INT_MAX)
            #"enable_etag": boolean                                     # true
        }
    ]
}
```

절과 설정 항목의 구성에 대한 기호나 내용에 대한 자세한 내용은 설정 항목 값의 형식 및 설정 방법을 참고합니다.

jeus

WebtoB - JEUS 간 연동하여 WJP로 서비스를 수행할 경우 설정합니다.

구분	설명
자료형	array(object)

jeus/name (필수 항목)

JEUS의 이름을 설정합니다. 해당 이름은 wjp/wjp_servers/name에 설정된 이름과 일치해야 합니다.

구분	설명
자료형	string
범위	31자 이내

jeus/server_schedule

wjp/wjp_servers에 서버를 여러 개 설정했을 경우 요청을 처리할 서버의 지정 방법을 설정합니다.

구분	설명
자료형	string
범위	"RR"
기본값	"RR"

다음은 설정값에 대한 설명입니다.

설정값	설명
RR	Round Robin 방식으로 순차적으로 JEUS에 요청을 할당합니다.

jeus/connection_schedule

JEUS 서버에 여러 개의 커넥션(Connection)이 맺어져 있을 경우 요청을 처리할 커넥션의 지정 방법을 설정합니다.

구분	설명
자료형	string
범위	"RR"
기본값	"RR"

다음은 설정값에 대한 설명입니다.

설정값	설명
RR	Round Robin 방식으로 순차적으로 커넥션에 요청을 할당합니다.

jeus/enable_flexible_sticky_session_routing

세션 라우팅 시 유연하게 라우팅할지 여부를 설정합니다.

구분	설명
자료형	boolean
기본값	false

다음은 설정값에 대한 설명입니다.

설정값	설명
true	sticky_session_routing_id 기준으로 세션 라우팅 시 내부 서버의 persistent_server_connections_max 모두 RUN 상태이면 큐잉(Queuing)하지 않고 다른 sticky_session_routing_id를 가진 내부 서버의 커넥션으로 라우팅합니다.
false	기본으로 스티키 세션 라우팅(Sticky Session Routing)이 됩니다. 이는 동일한 sticky_session_routing_id를 가진 내부 서버의 커넥션으로만 라우팅하고, 해당 내부 서버의 persistent_server_connections_max가 모두 RUN 상태이면 큐잉한다는 의미입니다.

유연한(Flexible) 라우팅을 하는 경우 동일한 JSESSIONID를 가진 다른 클라이언트의 요청이 각각 다른 서버로 라우팅될 수 있으므로 주의합니다. (**기본값 사용 권장**)

jeus/enable_request_level_ping

요청을 포워딩하기 전에 해당 서버에 PING을 보내는지 여부를 설정합니다. 여러 JEUS 서버가 연결된 경우 OOM(Out Of Memory)등으로 요청을 보내도 응답하기 어려운 서버가 발생할 수 있는 경우에 사용하면 유용합니다.

구분	설명
자료형	boolean
기본값	false

다음은 설정값에 대한 설명입니다.

설정값	설명
true	JEUS 서버로 포워딩하는 모든 요청에 대해 PING을 보내고 이에 대한 응답을 받은 경우에만 요청을 포워딩합니다.
false	JEUS 서버로 포워딩하는 모든 요청에 대해 PING을 보내지 않고 요청을 바로 포워딩합니다.

모든 요청에 대해 PING을 보내고 응답을 받아야 하므로 성능 저하가 발생할 수 있습니다.
(**기본값 사용 권장**)

jeus/request_level_ping_timeout

PING을 보내고 이에 대한 응답을 받기까지 기다리는 시간을 설정합니다. 해당 시간 안에 응답을 받지 못하면 해당 연결을 끊고 다른 JEUS 서버를 다시 스케줄링하여 PING을 보냅니다.

구분	설명
자료형	integer
단위	초
범위	0 ~ INT_MAX
기본값	3

jeus/request_level_ping_retry_count

JEUS 서버 내에 모든 Jengine에 대해 PING을 보내고 실패한 경우 재시도하는 횟수를 설정합니다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	0

jeus/headers

적용할 HEADERS 절 이름을 설정합니다.

구분	설명
자료형	array(string)
범위	15개 이내(255자 이내)

jeus/session_id_cookie_key

세션 라우팅으로 사용되는 HTTP 쿠키의 Key 이름을 설정합니다.

구분	설명
자료형	string
범위	255자 이내
기본값	"JSESSIONID"

jeus/rewrite_cookie_domain

내부 서버 응답의 쿠키 헤더 필드 값의 "domain=" 값을 수정합니다. "domain="에 지정된 domain string이 RewriteCookieDomain에 지정된 string과 일치할 경우 "domain=" 값을 사용자 요청의 domain으로 교체합니다.

구분	설명
자료형	string
범위	255자 이내
기본값	"JSESSIONID"

아래와 같이 설정하면 WebtoB의 domain이 webtob인 경우 원본 응답의 쿠키인 "jsessionid=abc, domain=internal.server.com"은 "jsessionid=abc, domain=webtob"로 변경됩니다.

```
"rewrite_cookie_domain": "internal.server.com"
```

jeus/rewrite_cookie_path

내부 서버 응답의 쿠키 헤더 필드 값의 "path=" 값을 수정합니다.

구분	설명
자료형	object

jeus/rewrite_cookie_path/from

"path=" 값에서 수정할 string을 설정합니다.

구분	설명
자료형	string
범위	255자 이내

jeus/rewrite_cookie_path/to

"path=" 값으로 수정될 string을 설정합니다.

구분	설명
자료형	string
범위	255자 이내

jeus/enable_cache

콘텐츠의 캐시 유무를 설정합니다.

구분	설명
자료형	boolean
기본값	false

jeus/cache_refresh

JEUS로부터 받은 응답을 캐시하게 되는 경우 캐시되는 응답의 유효 시간을 계산하기 위한 값을 설정합니다.

클라이언트의 요청에 대해 WebtoB가 캐시된 내용에서 응답할 때는 캐시된 내용이 유효한지를 판단합니다. 이때 유효 여부는 설정된 값(유효 시간)을 기준으로 판단합니다.

구분	설명
자료형	integer
단위	초
범위	1 ~ INT_MAX
기본값	3600

구분	설명
설정 우선순위	설정할 때 적용되는 우선순위는 다음과 같습니다. 1. Cache-Control:max-age 첫 번째로 캐시될 응답 헤더에 Cache-Control:max-age 값이 있는 경우 max-age(초) 동안 유효합니다. 2. Expires 두 번째로 캐시될 응답 헤더에 Expires 값이 있으면 Expires(시간)까지 유효합니다. 3. cache_refresh 세 번째로 응답 헤더에 'Cache-Control:max-age', 'Expires' 값이 없는 경우 JEUS로부터 받은 응답이 캐시된 후 캐시된 내용은 cache_refresh(초) 동안 유효합니다.

jeus/max_queue_count

클라이언트의 요청이 과도하게 증가하여 정상적인 서비스 처리가 어려울 경우 계속되는 서비스 요청을 무시할 필요가 있습니다.

큐에 쌓인 클라이언트의 요구 수가 어느 정도 이상이 되면 새로 도착한 요구는 큐에 쌓이지 않고 클라이언트에 즉시 에러로 응답합니다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	0

jeus/max_queue_url

서버 큐가 다 찼을 경우 대신 서비스할 페이지를 설정합니다.

구분	설명
자료형	string
범위	255자 이내

jeus/max_queue_status_code

서버 큐가 다 찼을 경우 응답할 상태 코드를 설정합니다.

구분	설명
자료형	string
범위	301 | 302 | 303 | 307 | 308 | 410 | 503
기본값	503

jeus/queue_timeout

서버 큐에 대기하고 있는 사용자 요청의 타임아웃을 설정합니다.

사용자 요청이 많아 해당 요청을 처리할 서버 프로세스가 없을 경우 해당 요청은 서버 큐에서 요청을 처리할 서버 프로세스가 생길 때까지 기다립니다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	0

다음은 설정값에 대한 설명입니다.

설정값	설명
0	서버 큐에서 대기하는 시간을 제한하지 않습니다.
양의 정수값	설정된 시간 이상 대기하는 요청은 큐에서 제거되며, "503 Service Unavailable"로 응답합니다.

다음은 max_queue 설정 예시입니다.

```
...
"destination": {
  "jeus": {
    "max_queue_count": 3,
    "max_queue_url":"/jsvtest/common/test.html",
    "max_queue_status_code":302,
    "queue_timeout":100
  }
},
...
```

위와 같이 설정하면 JEUS 연결이 2개일 때 10개의 호출을 한다면 2개 페이지 호출, 3개(queue) 페이지 호출, 5개 max_queue_url 호출로 응답합니다.

jeus/backup_server

서버들이 모두 준비 상태가 아닐 경우 백업 용도로 사용할 서버를 설정합니다.

구분	설명
자료형	string
범위	31자 이내

reverse_proxy

HTTP 프록시 일종으로 내부 서버로 요청을 전달할 때 설정하며, WAS 연동에 사용할 수 있습니다.

구분	설명
자료형	object

reverse_proxy/common_config

역방향 프록시의 공통적인 설정입니다. 상위 항목에 설정하면 하위 항목에는 설정하지 않아도 상위 항목의 설정이 반영됩니다.

구분	설명
자료형	object
설정 우선순위	설정할 때 적용되는 우선순위는 다음과 같습니다. 1. reverse_proxy_server 2. reverse_proxy_group 3. reverse_proxy

reverse_proxy/common_config/headers

적용할 HEADERS 절 이름을 설정합니다.

구분	설명
자료형	array(string)
범위	15개 이내(255자 이내)

reverse_proxy/common_config/compression

응답을 압축할 대상을 설정합니다. 압축할 MIME-Type을 설정하며, 응답의 Content-Type이 사용됩니다. 해당되는 응답은 클라이언트로 전송되기 전에 GZIP을 사용해서 압축됩니다.

구분	설명
자료형	array(string)
범위	32개 이내(255자 이내)

압축 기능을 사용할 경우 네트워크 트래픽을 줄일 수 있지만, 서버의 성능은 저하될 수 있습니다.

압축률이 낮은 파일(zip 같은 압축 파일이나 jpeg 같은 압축 이미지 등)을 압축할 경우 서버에 부하만 주기 때문에 해당 MIME-Type이 압축 대상이 되지 않도록 주의합니다.

압축 기능은 HTTP 요청 헤더 중 Accept-Encoding에 GZIP이나 deflate로 지정된 요청에 대해서만 적용됩니다.

reverse_proxy/common_config/compression_min_size

압축 설정을 통해 응답을 압축할 때 응답의 최소 크기를 설정합니다.

Content-Length 응답 헤더의 값이 설정된 값보다 크면 응답을 압축합니다. 단, chunked 응답인 경우 응답의 크기를 알기 어려우므로 적용되지 않습니다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	0

reverse_proxy/common_config/rewrite_redirect

내부 서버 응답에 "Location", "Content-Location" 헤더가 있는 경우 값을 수정합니다. 해당 필드가 "original_uri"로 시작하는 경우 "original_uri"와 일치하는 부분을 "redirect_path"로 교체합니다.

구분	설명
자료형	array(object)
범위	16개 이내(original_uri, redirect_path 각각 255자 이내)

아래와 같이 설정하면 원본 응답의 Location이 "http://internal.server.com:80/docs_kr/abc.html"인 경우 "/internal_kr/abc.html"로 변경됩니다.

```
"rewrite_redirect": [
    {
        "original_uri": "http://internal.server.com:80/docs/",
        "redirect_path": "/internal/"
    },
    {
        "original_uri": "http://internal.server.com:80/docs_kr",
        "redirect_path": "/internal_kr/"
    },
    {
        "original_uri": "http://internal.server.com:80/docs_ch/",
```

```
        "redirect_path": "/internal_ch/"
    }
]
```

reverse_proxy/common_config/rewrite_cookie_domain

내부 서버 응답의 쿠키 헤더 필드 값의 "domain=" 값을 수정합니다. "domain="에 지정된 domain string이 rewrite_cookie_domain에 지정된 string과 일치할 경우 "domain=" 값을 사용자 요청의 domain으로 교체합니다.

구분	설명
자료형	string
범위	255자 이내
기본값	0

아래와 같이 설정하면 WebtoB의 domain이 webtob인 경우 원본 응답의 쿠키인 "jsessionid=abc, domain=internal.server.com"은 "jsessionid=abc, domain=webtob"로 변경됩니다.

```
"rewrite_cookie_domain": "internal.server.com"
```

reverse_proxy/common_config/rewrite_cookie_path

내부 서버 응답의 쿠키 헤더 필드 값의 "path=" 값을 수정합니다.

구분	설명
자료형	object

아래와 같이 설정하면 원본 응답의 쿠키의 "jsessionid=abc, path=/jeus/application"이 "jsessionid=abc, path=/jeus_proxy/application"으로 변경됩니다.

```
"rewrite_cookie_path": {
    "from": "jeus",
    "to": "/jeus_proxy"
}
```

reverse_proxy/common_config/rewrite_cookie_path/from (필수 항목)

"path=" 값에서 수정할 string을 설정합니다.

구분	설명
자료형	string
범위	255자 이내

reverse_proxy/common_config/rewrite_cookie_path/to(필수 항목)

"path=" 값으로 수정될 string을 설정합니다.

구분	설명
자료형	string
범위	255자 이내

reverse_proxy/common_config/max_queue_count

클라이언트의 요청이 과도하게 증가하여 정상적인 서비스 처리가 어려울 경우 계속되는 서비스 요청을 무시할 필요가 있습니다.

큐에 쌓인 클라이언트의 요구 수가 어느 정도 이상이 되면 새로 도착한 요구는 큐에 쌓이지 않고 클라이언트에 즉시 에러로 응답합니다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	0

reverse_proxy/common_config/max_queue_url

서버 큐가 다 찼을 경우 대신 서비스할 페이지를 설정합니다.

구분	설명
자료형	string
범위	255자 이내

reverse_proxy/common_config/max_queue_status_code

서버 큐가 다 찼을 경우 응답할 상태 코드를 설정합니다.

구분	설명
자료형	integer
범위	301 | 302 | 303 | 307 | 308 | 410 | 503
기본값	503

reverse_proxy/common_config/websocket_connections_max

HTTP 커넥션에서 웹 소켓 커넥션으로 업그레이드 된 경우 해당 커넥션 수에 대한 제한을 설정합니다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	0

다음은 설정값에 대한 설명입니다.

설정값	설명
0	웹 소켓 커넥션 수를 제한하지 않습니다.

reverse_proxy/common_config/websocket_session_timeout

웹 소켓 커넥션에 대한 타임아웃을 설정합니다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	0

다음은 설정값에 대한 설명입니다.

설정값	설명
0	타임아웃을 체크하지 않습니다.

내부 서버의 웹 소켓 세션에 대한 타임아웃 설정과 맞춰서 설정할 것을 권장합니다.

reverse_proxy/common_config/set_host_header

역방향 프록시를 사용하여 내부 서버로 요청을 포워딩할 때 호스트 헤더의 값을 설정합니다.

설정하지 않으면 호스트 헤더의 값은 ServerAddress에 설정한 값이 설정되고, "$BypassHostHeader"로 설정하면 클라이언트가 보낸 호스트 헤더의 값을 그대로 사용합니다.

구분	설명
자료형	string
범위	255자 이내
기본값	0

reverse_proxy/common_config/queue_timeout

서버 큐에 대기하고 있는 사용자 요청의 타임아웃을 설정합니다.

사용자 요청이 많아 해당 요청을 처리할 서버 프로세스가 없을 경우 해당 요청은 서버 큐에서 요청을 처리할 서버 프로세스가 생길 때까지 기다립니다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	0

다음은 설정값에 대한 설명입니다.

설정값	설명
0	서버 큐에서 대기하는 시간을 제한하지 않습니다.
양의 정수값	설정된 시간 이상 대기하는 요청은 큐에서 제거되며, "503 Service Unavailable"로 응답합니다.

reverse_proxy/common_config/name_resolution_interval

Hostname Resolution을 수행할 주기를 설정합니다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	0

다음은 설정값에 대한 설명입니다.

설정값	설명
0	운영체제의 TTL(Time-To-Live) 설정을 따릅니다.

reverse_proxy/common_config/server_health_check

역방향 프록시 그룹의 헬스 체크 정보를 설정합니다.

구분	설명
자료형	object

다음은 server_health_check 사용 예시입니다.

```
  "destination": {
    "reverse_proxy ": {
      "common_config": {
        "server_health_check":{
            "retry_count":4,
            "failback_interval":5,
            "failure":["connection_timeout"],
            "enable_name_resolution_on_fail":false
          }
      },
```

reverse_proxy/common_config/server_health_check/retry_count

Failover를 판단할 재시도 횟수를 설정합니다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	3

reverse_proxy/common_config/server_health_check/failback_interval

Failover 이후 Failback 시도를 위해 커넥션을 시도할 헬스 체크 시간을 설정합니다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	60

다음은 설정값에 대한 설명입니다.

설정값	설명
0	Failover 및 Failback을 수행하지 않습니다.

reverse_proxy/common_config/server_health_check/failure

연결 실패를 판단할 기준을 설정합니다.

구분	설명
자료형	array(string)
범위	1~3600(초)
기본값	5(초)

reverse_proxy/common_config/server_health_check/enable_name_resolution_on_fail

Failure 판단 시 Hostname Resolution을 새로 시도할 지 여부를 설정합니다.

구분	설명
자료형	boolean
기본값	true

reverse_proxy/common_config/rewrite_html_url

HTML 페이지 응답 본문에 포함된 URL을 변경할 때 사용합니다. URL이 호스트를 포함할 경우 호스트는 요청에 사용된 WebtoB 서버 주소로 교체됩니다.

구분	설명
자료형	array(object)
범위	64개 이내

다음은 img tag의 src, longdesc, usemap attribute에 대해 http://test2:80으로 되어 있는 URL을 /proxy/로 변경하는 예시입니다.

```
"rewrite_html_url": [
    {
        "tag": "img",
        "attribute": ["src", "longdesc"],
        "from": "http://test2:80",
        "to": "/proxy/"
    }
]
```

reverse_proxy/common_config/rewrite_html_url/tag (필수 항목)

태그 이름을 설정합니다.

구분	설명
자료형	string
범위	255자 이내

reverse_proxy/common_config/rewrite_html_url/attribute (필수 항목)

속성 이름을 설정합니다.

구분	설명
자료형	array(string)

구분	설명
범위	64개 이내(255자 이내)

reverse_proxy/common_config/rewrite_html_url/from (필수 항목)

URL에서 수정할 string을 설정합니다.

구분	설명
자료형	string
범위	255자 이내

reverse_proxy/common_config/rewrite_html_url/to (필수 항목)

URL로 수정될 string을 설정합니다.

구분	설명
자료형	string
범위	255자 이내

reverse_proxy/common_config/rewrite_html_max_size

응답이 HTML 페이지인 경우(Content-Type: text/html) 페이지 내부의 특정 태그 값들을 수정할 수 있는데, 이때 사용되는 내부 버퍼의 최대 크기를 설정합니다. 만약 응답이 설정된 값보다 크면, 수정되지 않은 원본 응답이 클라이언트로 전송됩니다.

구분	설명
자료형	integer
범위	1 ~ INT_MAX
기본값	10240

reverse_proxy/reverse_proxy_group (필수 항목)

reverse_proxy 설정을 그룹으로 관리하며, 다중 서버를 구성할 수 있습니다.

reverse_proxy(내부 서버)를 여러 개 설정하여 그룹으로 묶어 로드 밸런싱 및 스티키 세션 라우팅(Sticky Session Routing)을 할 수 있으며, WAS 연동에 사용할 수 있습니다.

구분	설명
자료형	array(object)

reverse_proxy/reverse_proxy_group/name (필수 항목)

reverse_proxy_group 이름을 설정합니다.

구분	설명
자료형	string
범위	31자 이내

reverse_proxy/reverse_proxy_group/reverse_proxy_server (필수 항목)

하나의 reverse_proxy_group에서 ip:port 별로 각각 설정을 다르게 관리할 수 있습니다.

구분	설명
자료형	array(object)
범위	최대 32개 이내

reverse_proxy/reverse_proxy_group/reverse_proxy_server/name (필수 항목)

역방향 프록시 서버 이름을 설정합니다.

구분	설명
자료형	string
범위	31자 이내

reverse_proxy/reverse_proxy_group/reverse_proxy_server/address (필수 항목)

요청이 전달되는 내부 서버 주소를 설정합니다.

구분	설명
자료형	string
범위	255자 이내

reverse_proxy/reverse_proxy_group/reverse_proxy_server/enable_proxy_ssl

내부 서버와 SSL/TLS 프로토콜을 사용하여 연결할지 여부를 설정합니다. 이때 proxy_ssl_name 설정으로 적용할 PROXY_SSL 절 항목을 지정할 수 있습니다.

구분	설명
자료형	boolean
기본값	false

reverse_proxy/reverse_proxy_group/reverse_proxy_server/proxy_ssl_name

사용할 ssl/proxy_ssl_configs 절의 이름을 설정합니다. 단, enable_proxy_ssl을 'true'로 설정하는 경우 적용됩니다.

구분	설명
자료형	array(string)
범위	100개 이내(31자 이내)

다음은 proxy_ssl_name 사용 예시입니다.

```
"destination": {
  "reverse_proxy":{
    "reverse_proxy_server":[
      {
        "address":"internal.server.com:80",
        "name":"rproxy1",
        "enable_proxy_ssl": true,
        "proxy_ssl_name":["ssl1"]
      }
    ]
  }
}
```

reverse_proxy/reverse_proxy_group/reverse_proxy_server/persistent_server_connections_min

요청 처리 후 내부 서버와의 커넥션을 지속적으로 유지하고자 하는 경우 최소 개수를 설정합니다.

신규 요청이 들어오면 내부 서버와의 커넥션이 새로 맺어지고, 내부 서버가 커넥션을 끊지 않는 이상 해당 커넥션을 지속적으로 유지하며 이후 요청에 대하여 이 커넥션을 재사용합니다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	0

reverse_proxy/reverse_proxy_group/reverse_proxy_server/persistent_server_connections_max

요청 처리 후 내부 서버와의 커넥션을 지속적으로 유지하고자 하는 경우 최대 개수를 설정합니다.

재사용 가능한 커넥션이 모두 다른 요청을 처리 중이라면 내부 서버와의 커넥션을 새로 맺어서 추가하며, 이미 최대 개수가 재사용 중이라면 요청은 큐잉된다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	0

reverse_proxy/reverse_proxy_group/reverse_proxy_server/persistent_server_check_url

내부 서버와의 커넥션을 지속적으로 유지하기 위해 내부적으로 HTTP HEAD 요청을 PING 메시지로 사용합니다.

구분	설명
자료형	string
범위	255자 이내
기본값	"/"

HTTP 요청을 내부 서버로 보내 HTTP 응답을 받아 이를 PONG 메시지로 확인하고 커넥션을 지속적으로 유지합니다. 200 응답이 아닌 경우 커넥션은 지속적으로 유지되지 않으며 이를 위해 내부 서버의 PING 요청에 대해 응답할 수 있는 애플리케이션을 만들어야 합니다.

reverse_proxy/reverse_proxy_group/reverse_proxy_server/persistent_server_check_time

내부 서버와의 커넥션을 체크하는 시간을 설정하며 내부 서버와의 커넥션을 확인 및 지속적으로 유지하는 데 사용됩니다.

내부 서버와의 커넥션을 지속적으로 유지하기 위해서는 내부 서버의 keepalive_timeout보다 작게 설정합니다. 준비 상태의 커넥션에 대하여 한 번의 PING 메시지를 보내고 persistent_server_check_time이 지나도록 응답이 오지 않으면 해당 커넥션은 이상이 발생했다고 판단하여 커넥션을 끊습니다.

구분	설명
자료형	integer
단위	초
범위	0 ~ INT_MAX
기본값	30

다음은 설정값에 대한 설명입니다.

설정값	설명
0	PING 메시지를 보내지 않습니다. 이 경우 내부 서버의 keepalive_timeout만큼만 커넥션을 지속합니다.

reverse_proxy/reverse_proxy_group/reverse_proxy_server/persistent_server_timeout

내부 서버와의 커넥션이 persistent_server_connections_min보다 많을 경우 준비 상태가 지속되고 있는 커넥션에 대한 타임아웃을 설정합니다.

구분	설명
자료형	integer
단위	초
범위	0 ~ INT_MAX
기본값	300

다음은 설정값에 대한 설명입니다.

설정값	설명
0	유휴(Idle) 커넥션에 대한 타임아웃을 체크하지 않습니다.

다음은 persistent_server 사용 예시입니다.

```
"destination": {
  "reverse_proxy":{
    "reverse_proxy_group":[
      {
        "name": "rproxyg1",
        ...
        "persistent_server_connections_min":10,
        "persistent_server_connections_max":15,
        "persistent_server_check_time": 25,
        "persistent_server_check_url":"/jsvtest/common/test.html",
        "persistent_server_timeout":3000
      }
    ]
  }
}
```

reverse_proxy/reverse_proxy_group/reverse_proxy_server/enable_cache

콘텐츠의 캐시 유무를 설정합니다.

구분	설명
자료형	boolean
기본값	false

reverse_proxy/reverse_proxy_group/reverse_proxy_server/cache_refresh

역방향 프록시로 처리된 응답을 캐시하게 되는 경우 캐시되는 응답의 유효 시간을 계산하기 위한 값을 설정합니다.

클라이언트의 요청에 대해 WebtoB가 캐시된 내용에서 응답할 때는 캐시된 내용이 유효한지를 판단합니다. 이때 유효 여부는 설정된 값(유효 시간)을 기준으로 판단합니다.

구분	설명
자료형	integer
단위	초
범위	1 ~ INT_MAX
기본값	3600
설정 우선순위	설정할 때 적용되는 우선순위는 다음과 같습니다. 1. Cache-Control:max-age 첫 번째로 캐시될 응답 헤더에 Cache-Control:max-age 값이 있는 경우 max-age(초) 동안 유효합니다. 2. Expires 두 번째로 캐시될 응답 헤더에 Expires 값이 있으면 Expires(시간)까지 유효합니다. 3. cache_refresh 세 번째로 응답 헤더에 'Cache-Control:max-age', 'Expires' 값이 없는 경우 JEUS로부터 받은 응답이 캐시된 후 캐시된 내용은 cache_refresh(초) 동안 유효합니다.

reverse_proxy/reverse_proxy_group/reverse_proxy_server/sticky_session_routing_id

특정 내부 서버(WAS)를 연동하고, 스티키 세션 라우팅(Sticky Session Routing)을 사용할 경우 설정합니다.

내부 서버(WAS)에서 Set-Cookie 응답 헤더에 Sticky Session id(JSESSIONID)로 넣어주는 값의 엔진명을 사용합니다. 예를 들어 Set-Cookie 응답 헤더의 "JSESSIONID" 값이 "Pl1xfBkEVbUu2cj20CUNlHJoWLmU.xxx_servlet_engine1"이라면 점(dot)으로 구분하여 이후 값인 "xxx_servlet_engine1"을 입력합니다.

구분	설명
자료형	string
범위	255자 이내

reverse_proxy/reverse_proxy_group/reverse_proxy_server/is_backup_server

다른 역방향 프록시 서버들이 모두 준비 상태가 아닐 경우 백업 용도로 사용할 서버인지를 설정합니다.

구분	설명
자료형	boolean
기본값	false

reverse_proxy/reverse_proxy_group/reverse_proxy_server/load_balancing_factor

역방향 프록시 그룹에서 해당 역방향 프록시 서버에 요청을 분배할 비율을 설정합니다.

이때 "(역방향 프록시 서버의 load_balancing_factor 설정 값) / (그룹내 모든 역방향 프록시 서버의 load_balancing_factor 설정 값의 합)" 비율로 요청이 분배됩니다.

구분	설명
자료형	integer
범위	1 ~ INT_MAX
기본값	1

reverse_proxy/reverse_proxy_group/reverse_proxy_server/common_config

역방향 프록시의 공통적인 설정으로 상위 항목에 설정하면 하위 항목에는 설정하지 않아도 상위 항목의 설정이 반영됩니다.

구분	설명
자료형	object
설정 우선순위	설정할 때 적용되는 우선순위는 다음과 같습니다. 1. reverse_proxy_server 2. reverse_proxy_group 3. reverse_proxy

reverse_proxy/reverse_proxy_group/server_schedule

역방향 프록시를 여러 개 설정하였을 경우 요청을 처리할 역방향 프록시의 지정 방법을 설정합니다.

구분	설명
자료형	string
범위	"RR"
기본값	"RR"

다음은 설정값에 대한 설명입니다.

설정값	설명
RR	Round Robin 방식으로 순차적으로 역방향 프록시에 요청을 할당합니다.

reverse_proxy/reverse_proxy_group/sticky_session_routing

세션 라우팅과 관련된 설정입니다.

구분	설명
자료형	object

reverse_proxy/reverse_proxy_group/sticky_session_routing/policy

세션 라우팅 정책을 설정합니다.

구분	설명
자료형	string
범위	"AddNewCookie" | "ModifyOriginalCookie" | "UseOriginalCookie"
기본값	"UseOriginalCookie"

reverse_proxy/reverse_proxy_group/sticky_session_routing/session_id_cookie_key

세션 라우팅으로 사용되는 HTTP 쿠키의 Key 이름을 설정합니다.

구분	설명
자료형	string
기본값	"JSESSIONID"

reverse_proxy/reverse_proxy_group/sticky_session_routing/enable_flexible_sticky_session_routing

세션 라우팅 시 유연하게 라우팅할지 여부를 설정합니다.

구분	설명
자료형	boolean
기본값	false

다음은 설정값에 대한 설명입니다.

설정값	설명
true	sticky_session_routing_id를 기준으로 세션 라우팅할 때 내부 서버의 persistent_server_connections_max가 모두 RUN 상태이면 큐잉(Queuing)하지 않고 다른 sticky_session_routing_id를 가진 내부 서버의 커넥션으로 라우팅합니다.
false	기본으로 스티키 세션 라우팅(Sticky Session Routing)이 됩니다. 이는 동일한 sticky_session_routing_id를 가진 내부 서버의 커넥션으로만 라우팅하고, 해당 내부 서버의 persistent_server_connections_max가 모두 RUN 상태이면 큐잉한다는 의미입니다.

유연한(Flexible) 라우팅을 하는 경우 동일한 JSESSIONID를 가진 다른 클라이언트의 요청이 각각 다른 서버로 라우팅될 수 있으므로 주의합니다. (기본값 사용 권장)

reverse_proxy/reverse_proxy_group/common_config

역방향 프록시의 공통적인 설정으로 상위 항목에 설정하면 하위 항목에는 설정하지 않아도 상위 항목의 설정이 반영됩니다.

구분	설명
자료형	object
설정 우선순위	설정할 때 적용되는 우선순위는 다음과 같습니다. 1. reverse_proxy_server 2. reverse_proxy_group 3. reverse_proxy

htmls

정적 파일 요청을 처리할 경우 설정합니다. 이때 HTTP 메소드는 "GET", "POST", "HEAD"를 지원하며, 다른 메소드에 대해서는 "405 Method Not Allowed"로 응답합니다.

구분	설명
자료형	array(object)

htmls/name(필수 항목)

htmls의 이름을 설정합니다.

구분	설명
자료형	string
범위	31자 이내

htmls/headers

적용할 HEADERS 절 이름을 설정합니다.

구분	설명
자료형	array(string)
범위	15개 이내(255자 이내)

htmls/compression

정적 파일에 대한 응답에 대해 압축할 대상을 설정합니다. 압축할 MIME-Type을 설정하며, 응답의 Content-Type이 사용됩니다. 해당되는 응답은 클라이언트로 전송되기 전에 GZIP을 사용해서 압축됩니다.

구분	설명
자료형	array(string)
범위	32개 이내(255자 이내)

압축 기능을 사용할 경우 네트워크 트래픽을 줄일 수 있지만, 서버의 성능은 저하될 수 있습니다.

압축률이 낮은 파일(zip 같은 압축 파일이나 jpeg 같은 압축 이미지 등)을 압축할 경우 서버에 부하만 주기 때문에 해당 MIME-Type이 압축 대상이 되지 않도록 주의합니다.

압축 기능은 HTTP 요청 헤더 중 Accept-Encoding에 GZIP이나 deflate로 지정된 요청에 대해서만 적용됩니다.

htmls/compression_min_size

압축 설정을 통해 응답을 압축할 때 응답의 최소 크기를 설정합니다. 요청한 파일의 크기가 설정된 값보다 크면 응답을 압축합니다.

구분	설명
자료형	integer
단위	bytes
범위	0 ~ INT_MAX
기본값	0

htmls/enable_cache

콘텐츠의 캐시 유무를 설정합니다.

구분	설명
자료형	boolean
기본값	false

htmls/cache_refresh

캐시된 응답 중 "Content-Type"이 "text/html"인 응답에 대한 유효 시간을 설정합니다. 즉, "text/html" 응답은

캐시된 후 설정된 시간(초) 동안만 유효합니다.

구분	설명
자료형	integer
단위	초
범위	1 ~ INT_MAX
기본값	3600

Conditional-GET 요청은 캐시된 응답의 변경 여부를 체크합니다. 응답이 변경되면 캐시된 응답은 삭제 후 변경된 응답으로 갱신됩니다.

htmls/queue_timeout

사용자 요청이 많아 해당 요청을 처리할 서버 프로세스가 없을 경우 해당 요청은 서버 큐에서 요청을 처리할 서버 프로세스가 생길 때까지 기다립니다.

구분	설명
자료형	integer
단위	초
범위	1 ~ INT_MAX
기본값	0

다음은 설정값에 대한 설명입니다.

설정값	설명
0	서버 큐에서 대기하는 시간을 제한하지 않습니다.
양의 정수값	설정된 시간 이상 대기하는 요청은 큐에서 제거되며, "503 Service Unavailable"로 응답합니다.

htmls/enable_sendfile

sendfile 기능의 사용 여부를 설정합니다.

구분	설명
자료형	boolean
기본값	false

htmls/sendfile_min_size

요청한 파일의 크기가 설정된 값보다 크면 sendfile을 사용합니다.

구분	설명
자료형	integer
단위	bytes
범위	0 ~ INT_MAX
기본값	8192

htmls/enable_etag

정적 파일 처리 중 ETag의 사용 여부를 설정합니다.

구분	설명
자료형	boolean
기본값	true

다음은 설정값에 대한 설명입니다.

설정값	설명
false	WebtoB는 클라이언트로 전달되는 응답에 ETag를 추가하지 않고, 클라이언트의 요청에 포함된 ETag는 무시합니다.

2.4.2. 설정 예시

다음은 DESTINATION 절을 설정한 예시입니다.

```
"destination": {
    "jeus": [
        {
            "name": "MyGroup"
        }
    ],
    "reverse_proxy": {
        "common_config": {
                "server_health_check": {
                    "retry_count": 3,
                    "failback_interval": 5,
                    "failure" [connection_timeout, "http_invalid", "http_4xx"]
                }
        }
        "reverse_proxy_group": [
            {
                "name": "rproxy1",
                "reverse_proxy_server": [
                    {
                        "address": "192.168.15.114:28080",
                        "enable_proxy_ssl": false
                    }
                ]
```

```
            }
        ]
    }
    "htmls": [
        {
            "name": "htmls1"
        }
    ]
}
```

2.5. ERRORDOCUMENT 절

WebtoB에서 에러 문제가 발생했을 때 다음과 같은 4가지 방법으로 대응할 수 있습니다.

- 소스 코드에 정의된 에러 메시지를 출력
- 사용자가 정의한 에러 메시지를 출력
- 로컬 URL로 재전송
- 외부 URL로 재전송

2번째, 3번째, 4번째의 경우에는 ERRORDOCUMENT 절을 설정하여 특정 HTTP 응답 상태 코드 에 대해 특정 페이지로 리다이렉트합니다. HTTP 401 상태 코드를 제외한 HTTP 상태 코드를 모두 설정할 수 있습니다.

2.5.1. 설정 항목

다음은 DESTINATION 절의 환경 설정 형식입니다.

```
#"error_document": {
    #"error_document_list": [
        {
            "name": string,
            "status": integer,          # (HTTP status code)
            "url": string
        }
    ]
}
```

절과 설정 항목의 구성에 대한 기호나 내용에 대한 자세한 내용은 설정 항목 값의 형식 및 설정 방법을 참고합니다.

error_document_list

ERRORDOCUMENT 설정에 대한 목록입니다.

구분	설명
자료형	array(object)

error_document_list/name (필수 항목)

ERRORDOCUMENT 설정의 이름입니다. 다른 절에서 ERRRORDOCUMENT 절의 기능을 사용할 때는 이 'name'을 설정해야 합니다.

구분	설명
자료형	string
범위	31자 이내

error_document_list/status (필수 항목)

HTTP 상태 코드 값을 설정합니다.

구분	설명
자료형	integer
범위	100 ~ 599

error_document_list/url (필수 항목)

doc_root 이하의 상대 경로가 되거나 클라이언트가 해석할 수 있는 전체 경로 값을 설정합니다.

구분	설명
자료형	string
범위	255자 이내

2.5.2. 설정 예시

다음은 ERRORDOCUMENT 절을 설정한 예시입니다.

```
{
    "error_document": {
        "error_document_list": [
            {
                "name": "forbidden",
                "status": 403,
                "url": "err/403.html"
            },
            {
                "name": "notfound",
                "status": 404,
                "url": "http://tmaxsoft.co.kr/404.html"
```

```
            }
        ]
    }
}
```

2.6. FILTER 절

Filter 모듈을 사용할 FILTER 절을 정의하여, NODE 절이나 SERVER 절에서 filter 항목을 설정합니다.

2.6.1. 설정 항목

다음은 FILTER 절의 환경 설정 형식입니다.

```
#"filter": {
    #"filters": [
        {
            "name": string,
            "path": string,
            #"event": [string]
        }
    ]
}
```

절과 설정 항목의 구성에 대한 기호나 내용에 대한 자세한 내용은 설정 항목 값의 형식 및 설정 방법을 참고합니다.

filters

FILTER 설정에 대한 목록입니다.

구분	설명
자료형	array(object)

filters/name (필수 항목)

FILTER 설정의 이름입니다. 다른 절에서 FILTER 절의 기능을 사용할 때는 이 'name'을 설정해야 합니다.

구분	설명
자료형	string
범위	31자 이내

filters/url (필수 항목)

서버 안의 물리적 디렉터리의 Filter 모듈이 위치한 경로명을 설정합니다.

구분	설명
자료형	string
범위	255자 이내

filters/event

Filter 모듈이 동작하는 타이밍을 설정합니다.

구분	설명
자료형	array(string)
범위	15개 이내(255자 이내)

다음은 종류별 필터 이벤트에 대한 설명입니다. 필터 이벤트 종류에 따라 설정하는 위치가 NODE 절 또는 SERVER 절로 구분됩니다.

- SYSTEM 필터 이벤트

  SYSTEM 필터 이벤트를 포함하는 필터는 NODE 절에 설정합니다.

필터 이벤트	설명
ON_STOP	WebtoB 엔진 종료 시점에 동작합니다.
ON_START	WebtoB 엔진 기동 시점에 동작합니다.

- HTTP 필터 이벤트

  HTTP 필터 이벤트를 포함하는 필터는 SERVER 절에 설정합니다.

필터 이벤트	설명
RECEIVE_REQUEST	클라이언트에게 HTTP 요청을 받은 시점에 동작합니다.
BEFORE_SEND_REQUEST	클라이언트에게 받은 HTTP 요청을 내부 서버에 전달하기 전 시점에 동작합니다.
AFTER_SEND_REQUEST	클라이언트에게 받은 HTTP 요청을 내부 서버에 전달한 후 시점에 동작합니다.
RECEIVE_RESPONSE	내부 서버에게 HTTP 응답을 받은 시점에 동작합니다.
BEFORE_SEND_RESPONSE	내부 서버에게 받은 HTTP 응답을 클라이언트에 전달하기 전 시점에 동작합니다.
AFTER_SEND_RESPONSE	내부 서버에게 받은 HTTP 응답을 클라이언트에 전달한 후 점에 동작합니다.

2.6.2. 설정 예시

다음은 Filter를 구현하기 위한 설정 예시입니다.

- Filter 프로젝트 구조

  다음은 Filter를 구현하기 위한 프로젝트의 구조 예시입니다. 각 파일은 Filter를 정의하고 구현하는 데 필요한 요소들을 포함하고 있습니다.

  ```
  .
  ├─── CMakeLists.txt
  ├─── SampleSystemFilter.h
  ├─── SampleSystemFilter.cpp
  ├─── SampleHttpFilter.h
  └─── SampleHttpFilter.cpp
  ```

  각 파일의 설정 예시는 다음과 같습니다.

  - CMakeLists.txt

    ```
    cmake_minimum_required(VERSION 3.15)

    project(sampleFilter)
    set(CMAKE_VERBOSE_MAKEFILE true)
    set(CMAKE_CXX_STANDARD 17)

    message("\t MODULE: ${PROJECT_NAME}")

    # Webtob Filter Include Directory
    include_directories(/WEBTOB_BINARY_PATH/include)

    # Library Add
    add_library(sampleSystemFilter SHARED SampleSystemFilter.cpp)
    add_library(sampleHttpFilter SHARED SampleHttpFilter.cpp)
    ```

  - SampleSystemFilter.h

    ```
    /**
     * @File     : SampleFilter.h
     * @Author   : Webtob
     * @Year     : 2024
     */

    #ifndef WEBTOB6_SAMPLE_SYSTEM_FILTER_H
    #define WEBTOB6_SAMPLE_SYSTEM_FILTER_H

    #include <iostream>
    #include <string>
    #include <sstream>
    #include <memory>
    #include <vector>
    #include "webtob-filter/filter-api/WebtobFilter.h"
    ```

```
namespace webtob {
    class SampleSystemFilter : public WebtobFilter {
    public:
        static std::vector<std::string> calledEvents;

        FilterResult onStart() override;

        FilterResult onStop() override;

        FilterResult onReceiveRequest(const std::shared_ptr<FilterRequest>& request,
                                      const std::shared_ptr<FilterResponse>& response)
override;

        FilterResult onReceiveResponse(const std::shared_ptr<FilterRequest>& request,
                                       const std::shared_ptr<FilterResponse>& response)
override;

    private:
        static std::string getStaticVector();

        void printCalledInfo(const char* calledEvent);

        std::string filterName = "SampleSystemFilter";
    };
} // namespace webtob

#endif //WEBTOB6_SAMPLE_SYSTEM_FILTER_H
```

- SampleSystemFilter.cpp

```
#include "SampleSystemFilter.h"

using namespace webtob;

std::vector<std::string> SampleSystemFilter::calledEvents;

// Filter Creator
extern "C" [[maybe_unused]] std::shared_ptr<WebtobFilter> createFilter() {
    return std::make_shared<SampleSystemFilter>();
}

std::string SampleSystemFilter::getStaticVector() {
    std::ostringstream description;

    description << "Sample SystemFilter Event Called Vector= [\n";
    for (const std::string& event : calledEvents) {
        description << "\t" << event << "\n";
    }
    description << "]";
    return description.str();
}

void SampleSystemFilter::printCalledInfo(const char* calledEvent) {
    calledEvents.emplace_back(calledEvent);
    std::ostringstream message;
    message << "\n============================================";
    message << "\n" << filterName << " : " << calledEvent;
```

```
    message << "\n---------------------------------------------";
    message << "\n" << getStaticVector();
    message << "\n=============================================";
    std::cout << message.str() << std::endl;
}

FilterResult SampleSystemFilter::onStart() {
    printCalledInfo(__FUNCTION__);
    return FilterResult::SUCCESS;
}


FilterResult SampleSystemFilter::onStop() {
    printCalledInfo(__FUNCTION__);
    return FilterResult::SUCCESS;
}

FilterResult SampleSystemFilter::onReceiveRequest(const std::shared_ptr<FilterRequest>&
request,
                                                  const std::shared_ptr<FilterResponse>&
response) {
    printCalledInfo(__FUNCTION__);
    return FilterResult::SUCCESS;
}


FilterResult SampleSystemFilter::onReceiveResponse(const std::shared_ptr<FilterRequest>&
request,
                                                   const std::shared_ptr<FilterResponse>&
response) {
    printCalledInfo(__FUNCTION__);
    return FilterResult::SUCCESS;
}
```

- SampleHttpFilter.h

```
/**
 * @File     : SampleHttpFilter.h
 * @Author   : Webtob
 * @Year     : 2024
 */
#ifndef WEBTOB6_SAMPLE_HTTP_FILTER_H
#define WEBTOB6_SAMPLE_HTTP_FILTER_H

#include <iostream>
#include <sstream>
#include "webtob-filter/filter-api/WebtobFilter.h"

namespace webtob {
    class SampleHttpFilter : public WebtobFilter {
    public:
        FilterResult onReceiveRequest(const std::shared_ptr<FilterRequest>& request,
                                      const std::shared_ptr<FilterResponse>& response)
override;

        FilterResult onBeforeSendRequest(const std::shared_ptr<FilterRequest>& request,
                                         const std::shared_ptr<FilterResponse>& response)
```

```
override;

        FilterResult onAfterSendRequest(const std::shared_ptr<FilterRequest>& request,
                                        const std::shared_ptr<FilterResponse>& response)
override;

        FilterResult onReceiveResponse(const std::shared_ptr<FilterRequest>& request,
                                       const std::shared_ptr<FilterResponse>& response)
override;

        FilterResult onBeforeSendResponse(const std::shared_ref<FilterRequest>& request,
                                          const std::shared_ptr<FilterResponse>& response)
override;

        FilterResult onAfterSendResponse(const std::shared_ptr<FilterRequest>& request,
                                         const std::shared_ptr<FilterResponse>& response)
override;

        std::string filterName = "SampleHttpFilter";
    private:
        void printCalledInfo(const char* calledEvent);
    };
} // namespace webtob

#endif //WEBTOB6_SAMPLE_HTTP_FILTER_H
```

- SampleHttpFilter.cpp

```
#include "SampleHttpFilter.h"
#include <string_view>

using namespace webtob;

// Filter Creator
extern "C" [[maybe_unused]] std::shared_ptr<WebtobFilter> createFilter() {
    return std::make_shared<SampleHttpFilter>();
}


bool startsWith(const std::string_view& str, const std::string_view& prefix) {
    return str.substr(0, prefix.size()) == prefix;
}

void SampleHttpFilter::printCalledInfo(const char* calledEvent) {
    std::ostringstream message;
    message << "\n==============================================";
    message << "\n" << filterName << " : " << calledEvent;
    message << "\n==============================================";
    std::cout << message.str() << std::endl;
}

FilterResult SampleHttpFilter::onReceiveRequest(const std::shared_ptr<FilterRequest>&
request,
                                                const std::shared_ptr<FilterResponse>&
response) {

    printCalledInfo(__FUNCTION__);
```

```
    std::string path = request->getPath();
    if (startsWith(path, "/service/")) {
        if (startsWith(path, "/service/A")) {
            response->setStatusCode(200);
            response->setBody("Filter makes service A response");
        } else if (startsWith(path, "/service/B")) {
            response->setStatusCode(200);
            response->setBody("Service B response");
        } else if (startsWith(path, "/service/C")) {
            response->setStatusCode(400);
            response->setBody("BAD Response Makes");
        }
        return FilterResult::TERMINATE;
    }

    auto queryString = request->getQueryString();

    if (queryString == "test=reject1") {
        response->setStatusCode(200);
        response->setBody("I decide to reject");
        response->addHeader("ErrorCheck", "Reject1_Detected");
        return FilterResult::TERMINATE;
    }
    request->addHeader("Filter", "InsertOn");
    request->addCookie("FilterCookieKey", "OK_Inserted");
    return FilterResult::SUCCESS;
}

FilterResult SampleHttpFilter::onBeforeSendRequest(const std::shared_ptr<FilterRequest>&
request,
                                                   const std::shared_ptr<FilterResponse>&
response) {
    printCalledInfo(__FUNCTION__);
    auto queryString = request->getQueryString();
    if (queryString == "test=reject2") {
        response->setStatusCode(200);
        response->setBody("I decide to reject onBeforeSendRequest");
        response->addHeader("ErrorCheck", "onBeforeSendRequest");
        return FilterResult::TERMINATE;
    }

    return FilterResult::SUCCESS;
}

FilterResult SampleHttpFilter::onAfterSendRequest(const std::shared_ptr<FilterRequest>&
request,
                                                  const std::shared_ptr<FilterResponse>&
response) {
    printCalledInfo(__FUNCTION__);
    auto queryString = request->getQueryString();
    if (queryString == "test=reject3") {
        response->addHeader("ErrorCheck", "onAfterSendRequest");
        return FilterResult::TERMINATE;
    }
    return FilterResult::SUCCESS;
}

FilterResult SampleHttpFilter::onReceiveResponse(const std::shared_ptr<FilterRequest>&
```

```
request,
                                                    const std::shared_ptr<FilterResponse>&
response) {
    printCalledInfo(__FUNCTION__);
    auto cookieBuilder = response->getCookieBuilder();
    auto cookie = cookieBuilder-
>newCookie().setName("Cookie_ReceiveResponse").setValue("AddFromResponse")
        .setHttpOnly(true).setPath("/").setSameSite(FilterCookie::SameSite::OFF).build();
    response->addCookie(cookie);
    response->addHeader("Filter_response", "OK_ADDED");
    return FilterResult::SUCCESS;
}

FilterResult SampleHttpFilter::onBeforeSendResponse(const std::shared_ptr<FilterRequest>&
request,
                                                      const std::shared_ptr<FilterResponse>&
response) {
    printCalledInfo(__FUNCTION__);
    auto cookieBuilder = response->getCookieBuilder();
    auto cookie = cookieBuilder->newCookie().setName("MY_COOKIE").setValue("AddFromResponse")
        .setHttpOnly(true).setPath("/").setSameSite(FilterCookie::SameSite::OFF).build();
    response->addCookie(cookie);

    return FilterResult::SUCCESS;
}

FilterResult SampleHttpFilter::onAfterSendResponse(const std::shared_ptr<FilterRequest>&
request,
                                                     const std::shared_ptr<FilterResponse>&
response) {
    printCalledInfo(__FUNCTION__);
    return FilterResult::SUCCESS;
}
```

- 빌드

```
~/workspace/myTestFilter> cmake . -B build
-- The C compiler identification is AppleClang 15.0.0.15000309
-- The CXX compiler identification is AppleClang 15.0.0.15000309
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working C compiler: /Library/Developer/CommandLineTools/usr/bin/cc - skipped
-- Detecting C compile features
-- Detecting C compile features - done
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: /Library/Developer/CommandLineTools/usr/bin/c++ - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
     MODULE: sampleFilter
-- Configuring done (0.5s)
-- Generating done (0.0s)
-- Build files have been written to: /Users/seungwook/workspace/myTestFilter/build
~/workspace/myTestFilter>

~/workspace/myTestFilter> cd build && make
```

- 환경 설정

  다음 환경 설정 파일에서 FILTER 절을 설정하는 예시입니다.

```
{
    "filter": {
        "filters": [
            {
                "name": "filter1",
                "path": "TargetRealPath/myTestFilter/libsampleSystemFilter.so",
                "event": [ "ON_START", "ON_STOP" ]
            },
            {
                "name": "filter2",
                "path": "TargetRealPath/myTestFilter/libsampleHttpFilter.so",
                "event": [ "RECEIVE_REQUEST", "RECEIVE_RESPONSE" ]
            },
        ]
    }
}
```

2.7. HEADERS 절

사용자 요청 및 응답에 특정 HTTP 헤더를 변경하는 경우 HEADERS 절을 정의하여, SERVER 절이나 DESTINATION 절에 headers 항목을 설정합니다.

2.7.1. 설정 항목

다음은 HEADERS 절의 환경 설정 형식입니다.

```
#"headers": {
    #"headers_list": [
        {
            "name": string,
            "action": string,
            "field_name": string,
            #"field_value": string,
            #"reg_exp": string,
            #"status_code": integer     # 0
        }
    ]
}
```

절과 설정 항목의 구성에 대한 기호나 내용에 대한 자세한 내용은 설정 항목 값의 형식 및 설정 방법을 참고합니다.

headers_list

HEADERS 설정에 대한 목록입니다.

구분	설명
자료형	array(object)

headers_list/name (필수 항목)

HEADERS 설정의 이름입니다. 다른 절에서 HEADERS 절의 기능을 사용할 때는 이 'name'을 설정해야 합니다.

구분	설명
자료형	string
범위	255자 이내

headers_list/action (필수 항목)

HTTP 헤더의 처리 방식에 대한 동작을 설정합니다.

구분	설명
자료형	string
범위	"AddRequest" | "AddResponse" | "AddIfAbsentRequest" | "AddIfAbsentResponse" | "AppendResponse" | "EchoResponse" | "SetResponse" | "UnsetResponse"

다음은 각 동작에 대한 설명입니다.

종류	설명
AddRequest	요청에 설정된 HTTP 헤더를 추가합니다. 설정된 헤더가 요청에 이미 존재하는 경우 같은 헤더가 추가되기 때문에 사용에 유의합니다
AddResponse	응답에 설정된 HTTP 헤더를 추가합니다 설정된 헤더가 응답에 이미 존재하는 경우 같은 헤더가 추가되기 때문에 사용에 유의합니다
AddIfAbsentRequest	요청에 설정된 HTTP 헤더가 없는 경우 헤더를 추가합니다
AddIfAbsentResponse	응답에 설정된 HTTP 헤더가 없는 경우 헤더를 추가합니다
AppendResponse	이미 존재하는 헤더의 값 뒤에 설정된 FieldValue를 추가합니다. 해당 헤더가 없을 경우 아무런 동작도 수행하지 않습니다.
EchoResponse	요청에 설정된 HTTP 헤더가 존재하는 경우 응답에 같은 헤더 및 값을 추가합니다. 해당 동작은 FieldValue를 설정하지 않습니다.
SetResponse	응답에 설정된 HTTP 헤더가 없는 경우 헤더를 추가합니다. 이미 존재하는 경우 헤더의 값을 설정된 FieldValue로 치환합니다.

종류	설명
UnsetResponse	응답에 설정된 HTTP 헤더가 존재하는 경우 해당 헤더를 삭제합니다. 해당 동작은 FieldValue를 설정하지 않습니다.

동작의 종류에 따라 field_value는 필수 항목일 수 있습니다.

headers_list/field_name (필수 항목)

제어할 HTTP 헤더의 이름을 설정합니다.

구분	설명
자료형	string
범위	255자 이내

headers_list/field_value

제어할 HTTP 헤더의 값을 설정합니다.

구분	설명
자료형	string
범위	255자 이내

headers_list/reg_exp

HTTP 요청 URL이 Regular expression 패턴과 매칭되는 경우 헤더를 제어하기 위해 설정합니다.

구분	설명
자료형	string
범위	511자 이내

headers_list/status_code

특정 HTTP 상태 코드일 경우 헤더를 제어하기 위해 설정합니다.

구분	설명
자료형	integer
범위	0 ~ 599
기본값	0

2.7.2. 설정 예시

다음은 HEADERS 절을 설정한 예시입니다.

```
{
    "headers": {
    "headers_list": [
      {
        "name": "header1",
        "action": "AppendResponse",
        "field_name": "Test_Header",
        "field_value": "test"
      }
    ]
  }
}
```

2.8. LOGGING 절

클라이언트의 요구 내역을 기록하는 형식을 지정합니다. 접근 내역과 에러 내역이 따로 저장되며 저장 형식을 지정할 수 있습니다. 시스템 로그, 액세스 로그, 에러 로그 모두 LOGGING 절에 설정합니다.

2.8.1. 설정 항목

다음은 LOGGING 절의 환경 설정 형식입니다.

```
"logging": {
    "system_log": [
        {
            "name": string,
            #"level": string,                               # "INFO"
            #"dump": [string],
            #"handlers": {
                #"file_handler": {
                    "file_name": string,                    # "webtob_system.log"
                    #"rotate_by_seconds": integer,          # 0
                    #"valid_hours": integer,                # 0
                    #"rotate_by_file_size": integer,        # 0
                    #"archive_file_name": string,
                    #"enable_sync": boolean,                # false
                    #"permission": string                   # "0600"
                },
                #"enable_console_handler": boolean          # false
            }
        }
    ],
    #"access_log": [
        {
            "name": string,
            #"level": string,                               # "INFO"
            #"format": string,                              # "DEFAULT"
```

```
            #"exclude_by_ext": string,
            #"handlers": {
                #"file_handler": {
                    "file_name": string,                    # "webtob_access.log"
                    #"rotate_by_seconds": integer,          # 0
                    #"valid_hours": integer,                # 0
                    #"rotate_by_file_size": integer,        # 0
                    #"archive_file_name": string,
                    #"enable_sync": boolean,                # false
                    #"permission": string                   # "0600"
                },
                #"enable_console_handler": boolean          # false
            }
        }
    ],
    #"error_log": {
        #"level": "string",                                 # "INFO"
        #"format": "string",                                # "ERROR"
        #"enable_exclude_client_address_on_error": boolean, # false
        #"handlers": {
            #"file_handler": {
                "file_name": string,                        # "webtob_error.log"
                #"rotate_by_seconds": integer,              # 0
                #"valid_hours": integer,                    # 0
                #"rotate_by_file_size": integer,            # 0
                #"archive_file_name": string,
                #"enable_sync": boolean,                    # false
                #"permission": string                       # "0600"
            },
            #"enable_console_handler": boolean              # false
        }
    },
}
```

절과 설정 항목의 구성에 대한 기호나 내용에 대한 자세한 내용은 설정 항목 값의 형식 및 설정 방법을 참고합니다.

system_log (필수 항목)

시스템 로그를 설정합니다.

구분	설명
자료형	array(object)

system_log/name (필수 항목)

설정할 대상 시스템 로거를 지정합니다.

구분	설명
자료형	object

로거 이름은 '.'을 기준으로 하는 계층 구조로 구성되며, 로거에 대한 설정은 가장 하위 설정이 우선 적용됩니다. 예를 들어 (A)'webtob'와 (B)'webtob.http'가 모두 설정되어 있을 경우 'webtob.http' 로거는 (B) 설정을 따르고, 나머지 로거는 (A) 설정을 따릅니다. 최상위 시스템 로거인 'webtob'는 필수로 설정해야 합니다.

다음은 설정 가능한 로거에 대한 설명입니다.

로거 이름	적용 범위
webtob.http	HTTP 메시지를 다루는 전반에 대한 로그를 관리합니다.
webtob.wjp	WJP 메시지를 다루는 전반에 대한 로그를 관리합니다.
webtob.ssl	SSL 관련 로그를 관리합니다.
webtob.network	네트워크 전반에 대한 로그를 관리합니다.
webtob.server	각 스레드의 기동과 종료 관련 로그를 관리합니다.
webtob.em.acceptor	새로운 연결을 관리하는 Acceptor 관련 로그를 관리합니다.
webtob.em.control	Admin API 통신 관련 로그를 관리합니다.
webtob.em.worker	HTH 관련 로그를 관리합니다.
webtob.util	기타 유틸 함수 관련 로그를 관리합니다.

다음은 설정 예시입니다.

```
    "system_log": [
      {
        "name": "webtob",
        "level": "DEBUG",
        "handlers": {
          "file_handler": {
            "file_name": "webtob.log"
          },
          "enable_console_handler": false
        },
        "dump": [
          "ToClient",
          "FromClient",
          "ToServer",
          "FromServer",
          "SSLRead",
          "SSLWrite"
        ]
      },
      {
        "name":"webtob.http",
        "level":"DEBUG",
        "handlers": {
          "file_handler": {
            "file_name": "webtob_http.log"
          },
          "enable_console_handler": false
        }

      }
```

```
    ]
```

system_log/level

로그 레벨을 설정합니다.

시스템 로그 메시지의 레벨이 설정된 값 이상이면 출력됩니다. 레벨이 낮은 순서대로 "TRACE", "DEBUG", "INFO", "WARNING", "FATAL", "OFF"를 지원합니다.

구분	설명
자료형	string
범위	"TRACE" | "DEBUG" | "INFO" | "WARNING" | "FATAL" | "OFF"
기본값	"INFO"

레벨이 낮을수록 자세한 로그 메시지들이 출력됩니다.

system_log/dump

덤프(dump) 로그 출력 여부를 설정합니다. 이 설정은 로그 레벨이 DEBUG 또는 TRACE일 때만 적용되며, 이름이 "webtob"인 로거에 대해서만 설정할 수 있습니다.

구분	설명
자료형	array(string)
범위	6개 이내

다음은 설정값에 대한 설명입니다.

설정값	설명
FromClient	클라이언트가 보낸 데이터를 로그에 출력합니다.
ToClient	클라이언트로 전송된 데이터를 로그에 출력합니다.
FromServer	서버가 보낸 데이터를 로그에 출력합니다.
ToServer	서버로 전송된 데이터를 로그에 출력합니다.
SSLRead	SSL 암호화되어 읽혀진 데이터를 로그에 출력합니다.
SSLWrite	SSL 암호화되어 쓰여질 데이터를 로그에 출력합니다.

system_log/handlers

로그 메세지를 기록할 파일 및 콘솔에 대해 설정합니다.

구분	설명
자료형	object

system_log/handlers/file_handler

로그를 기록할 파일에 대해 설정합니다.

구분	설명
자료형	object

system_log/handlers/file_handler/file_name (필수 항목)

시스템 로그를 저장할 파일의 경로를 설정합니다. 상대 경로는 /로 시작하지 않으면 자동으로 "$WEBTOB6_HOME_PATH/상대 경로"로 대체됩니다.

구분	설명
자료형	string
범위	255자 이내
기본값	"webtob_system.log"

파일 이름에 포함된 다음의 대체 문자열은 파일 생성 시 실제 값으로 변경됩니다. 예를 들어 FileName에 "log/system_%Y%%M%%D%.log"로 설정된 값은 로그 파일 생성 시 "$WEBTOB6_HOME_PATH/log/system_20231023.log"로 변환될 수 있습니다.

다음은 대체 문자열에 대한 설명입니다.

문자열	실제값
%Y%	년도(year)로 숫자 4개를 설정합니다. (예: 2009)
%M%	월(month)로 숫자 2개를 설정합니다. (예: 11)
%D%	일(day)로 숫자 2개를 설정합니다. (예: 05)
%h%	시(hour)로 숫자 2개를 설정합니다. (예: 10)
%m%	분(minute)으로 숫자 2개를 설정합니다. (예: 30)
%s%	초(second)로 숫자 2개를 설정합니다. (예: 45)

system_log/handlers/file_handler/rotate_by_seconds

새로운 로그 메시지 생성할 때 현재 존재하는 로그 파일이 지정된 시간보다 오래되면 새로운 로그 파일을 생성하도록 설정합니다.

구분	설명
자료형	integer

구분	설명
범위	0 ~ 86400
기본값	0

system_log/handlers/file_handler/valid_hours

새로운 로그 파일을 설정된 시간 단위로 생성하도록 설정합니다.

구분	설명
자료형	integer
범위	0 ~ 23
기본값	0

system_log/handlers/file_handler/rotate_by_file_size

새로운 로그 메시지 생성할 때 현재 존재하는 로그 파일의 크기가 지정된 크기보다 큰 경우 새로운 로그 파일을 생성하도록 설정합니다.

구분	설명
자료형	integer
단위	bytes
범위	0 ~ INT_MAX
기본값	0

system_log/handlers/file_handler/archive_file_name

새로운 로그 파일 생성 시점에 기존 로그 파일은 archive_file_name에 설정된 이름으로 변경됩니다. 이후 새로운 로그 파일은 file_name에 설정된 형식으로 생성됩니다.

구분	설명
자료형	string
범위	255자 이내

system_log/handlers/file_handler/enable_sync

로그가 기록될 때 파일에 즉시 기록할지 여부를 설정합니다.

구분	설명
자료형	boolean
기본값	false

다음은 설정값에 대한 설명입니다.

설정값	설명
true	로그 메시지들이 WebtoB 메모리에 버퍼링되지 않고 바로 파일에 기록됩니다. **[참고]** 로그를 바로 확인해야 하는 곳에서는 해당 옵션을 설정하면 문제가 발생하는 경우 쉽게 확인할 수 있습니다.

system_log/handlers/file_handler/permission

시스템 로그 파일의 접근 권한을 설정합니다.

UNIX 계열 운영체제에서 사용하는 파일 접근 권한과 동일한 의미를 가지며, UNIX/Linux 환경에서만 사용할 수 있습니다.

구분	설명
자료형	string
범위	"0600" ~ "0777"

system_log/handlers/enable_console_handler

콘솔 창에 해당 로그를 출력할지 여부를 설정합니다.

구분	설명
자료형	boolean
범위	false

access_log (필수 항목)

액세스 로그를 설정합니다.

구분	설명
자료형	array(object)

access_log/name

액세스 로거의 이름을 설정합니다. 해당 이름은 HTTP 절 등에서 액세스 로그를 남길 대상을 지정할 때 사용됩니다.

구분	설명
자료형	object

access_log/level

액세스 로그 레벨을 설정합니다.

구분	설명
자료형	string
범위	"INFO" | "OFF"
기본값	"INFO"

다음은 설정값에 대한 설명입니다.

설정값	설명
OFF	액세스 로그를 출력하지 않습니다.

access_log/format

액세스 로그 파일에 기록될 메시지의 포맷을 설정합니다.

구분	설명
자료형	string
범위	255자 이내

다음은 설정값에 대한 설명입니다.

설정값	설명
DEFAULT	기본(Default) 로그 파일 포맷입니다. (로그 포맷: "%h %t \"%r\" %s %b %D")
COMMON	공통(Common) 로그 파일 포맷입니다. (로그 포맷: "%h %l %u %t \"%r\" %s %b")
COMBINED	결합(Combined) 로그 파일 포맷입니다. (로그 포맷: "%h %l %u %t \"%r\" %s %b \"%{*Referer*}i\" \"%{*User-Agent*}i\"")
COMBINEDIO	결합IO(CombinedIO) 로그 파일 포맷입니다. (로그 포맷: "%h %l %u %t \"%r\" %s %b \"%{*Referer*}i\" \"%{*User-Agent*}i\" %I %O")
ERROR	에러 로그 파일 포맷입니다. (로그 포맷: "%r")
%a	요청을 보낸 장비의 IP 주소를 표시합니다. (%h와 동일)
%A	서버의 IP 주소를 표시합니다.
%b	헤더를 제외한 응답의 byte를 표시합니다.

설정값	설명
%{_attr_name_}C	HTTP 요청의 쿠키 헤더 값 중 '_attr_name_'에 해당하는 값을 표시합니다.
%d	응답이 전송된 시간을 표시합니다.
%D	요청을 처리하는데 소요된 시간을 표시합니다. (단위: millisecond)
%{ENV_NAME}e	환경 변수 ENV_NAME을 출력합니다.
%h	요청을 보낸 장비의 IP 주소를 표시합니다. (%a와 동일)
%H	사용한 HTTP 버전을 표시합니다.
%{HEADER_FIELD}i	HTTP 요청의 HEADER_FIELD 헤더 값을 표시합니다.
%I	요청의 byte를 표시합니다.
%l	원격 로그인명을 표시합니다.
%m	HTTP 요청 메소드를 표시합니다.
%O	응답의 byte를 표시합니다.
%p	요청이 도착한 서버의 포트 번호를 표시합니다.
%q	HTTP 요청의 쿼리 값을 표시합니다.
%r	HTTP 요청의 Request line 전체를 표시합니다.
%s	응답에 사용된 HTTP 상태 코드를 표시합니다.
%S	http와 https를 구분하여 표시합니다.
%t	요청 처리를 마친 시간을 표시합니다.
%T	요청을 처리하는 데 소요된 시간을 표시합니다. (단위: 초)
%u	HTTP 인증에 사용된 사용자 이름을 표시합니다.
%U	HTTP 요청 URI를 표시합니다.
%v	호스트 헤더 필드 값을 표시합니다.

access_log/handlers

로그 메세지를 기록할 파일 및 콘솔에 대해 설정합니다.

구분	설명
자료형	object

access_log/handlers/file_handler

로그를 기록할 파일에 대해 설정합니다.

구분	설명
자료형	object

access_log/handlers/file_handler/file_name (필수 항목)

액세스 로그를 저장할 파일의 경로를 설정합니다. 상대 경로는 /로 시작하지 않으면 자동으로 "$WEBTOB6_HOME_PATH/상대 경로"로 대체됩니다.

구분	설명
자료형	string
범위	255자 이내
기본값	"webtob_access.log"

파일 이름에 포함된 다음의 대체 문자열은 파일 생성 시 실제 값으로 변경됩니다. 예를 들어 FileName에 "log/access_%Y%%M%%D%.log"로 설정된 값은 로그 파일 생성 시 "$WEBTOB6_HOME_PATH/log/access_20231023.log"로 변환될 수 있습니다.

다음은 대체 문자열에 대한 설명입니다.

문자열	실제값
%Y%	년도(year)로 숫자 4개를 설정합니다. (예: 2009)
%M%	월(month)로 숫자 2개를 설정합니다. (예: 11)
%D%	일(day)로 숫자 2개를 설정합니다. (예: 05)
%h%	시(hour)로 숫자 2개를 설정합니다. (예: 10)
%m%	분(minute)으로 숫자 2개를 설정합니다. (예: 30)
%s%	초(second)로 숫자 2개를 설정합니다. (예: 45)

access_log/handlers/file_handler/rotate_by_seconds

새로운 로그 메시지 생성할 때 현재 존재하는 로그 파일이 지정된 시간보다 오래되면 새로운 로그 파일을 생성하도록 설정합니다.

구분	설명
자료형	integer
범위	0 ~ 86400
기본값	0

access_log/handlers/file_handler/valid_hours

새로운 로그 파일을 설정된 시간 단위로 생성하도록 설정합니다.

구분	설명
자료형	integer
범위	0 ~ 23

구분	설명
기본값	0

access_log/handlers/file_handler/rotate_by_file_size

새로운 로그 메시지 생성할 때 현재 존재하는 로그 파일의 크기가 지정된 크기보다 큰 경우 새로운 로그 파일을 생성하도록 설정합니다.

구분	설명
자료형	integer
단위	bytes
범위	0 ~ INT_MAX
기본값	0

access_log/handlers/file_handler/archive_file_name

새로운 로그 파일 생성 시점에 기존 로그 파일은 archive_file_name에 설정된 이름으로 변경됩니다. 이후 새로운 로그 파일은 file_name에 설정된 형식으로 생성됩니다.

구분	설명
자료형	string
범위	255자 이내

access_log/handlers/file_handler/enable_sync

로그가 기록될 때 파일에 즉시 기록할지 여부를 설정합니다.

구분	설명
자료형	boolean
범위	false

다음은 설정값에 대한 설명입니다.

설정값	설명
true	로그 메시지들이 WebtoB 메모리에 버퍼링되지 않고 바로 파일에 기록됩니다. **[참고]** 로그를 바로 확인해야 하는 곳에서는 해당 옵션을 설정하면 문제가 발생하는 경우 쉽게 확인할 수 있습니다.

access_log/handlers/file_handler/permission

액세스 로그 파일의 접근 권한을 설정합니다.

UNIX 계열 운영체제에서 사용하는 파일 접근 권한과 동일한 의미를 가지며, UNIX/Linux 환경에서만 사용할 수 있습니다.

구분	설명
자료형	string
범위	"0600" ~ "0777"

access_log/handlers/enable_console_handler

콘솔 창에 해당 로그를 출력할지 여부를 설정합니다.

구분	설명
자료형	boolean
기본값	false

error_log (필수 항목)

에러 로그를 설정합니다.

구분	설명
자료형	array(object)

error_log/level

에러 로그 레벨을 설정합니다.

구분	설명
자료형	string
범위	"INFO" | "OFF"
기본값	"INFO"

다음은 설정값에 대한 설명입니다.

설정값	설명
OFF	에러 로그를 출력하지 않습니다.

error_log/format

에러 로그 파일에 기록될 메시지의 포맷을 설정합니다.

구분	설명
자료형	string
범위	255자 이내

다음은 설정값에 대한 설명입니다.

설정값	설명
DEFAULT	기본(Default) 로그 파일 포맷입니다. (로그 포맷: "%h %t \"%r\" %s %b %D")
COMMON	공통(Common) 로그 파일 포맷입니다. (로그 포맷: "%h %l %u %t \"%r\" %s %b")
COMBINED	결합(Combined) 로그 파일 포맷입니다. (로그 포맷: "%h %l %u %t \"%r\" %s %b \"%{*Referer*}i\" \"%{*User-Agent*}i\"")
COMBINEDIO	결합IO(CombinedIO) 로그 파일 포맷입니다. (로그 포맷: "%h %l %u %t \"%r\" %s %b \"%{*Referer*}i\" \"%{*User-Agent*}i\" %I %O")
ERROR	에러 로그 파일 포맷입니다. (로그 포맷: "%r")
%a	요청을 보낸 장비의 IP 주소를 표시합니다. (%h와 동일)
%A	서버의 IP 주소를 표시합니다.
%b	헤더를 제외한 응답의 byte를 표시합니다.
%{_attr_name_}C	HTTP 요청의 쿠키 헤더 값 중 '_attr_name_'에 해당하는 값을 표시합니다.
%d	응답이 전송된 시간을 표시합니다.
%D	요청을 처리하는데 소요된 시간을 표시합니다. (단위: millisecond)
%{ENV_NAME}e	환경 변수 ENV_NAME을 출력합니다.
%h	요청을 보낸 장비의 IP 주소를 표시합니다. (%a와 동일)
%H	사용한 HTTP 버전을 표시합니다.
%{HEADER_FIELD}i	HTTP 요청의 HEADER_FIELD 헤더 값을 표시합니다.
%I	요청의 byte를 표시합니다.
%l	원격 로그인명을 표시합니다.
%m	HTTP 요청 메소드를 표시합니다.
%O	응답의 byte를 표시합니다.
%p	요청이 도착한 서버의 포트 번호를 표시합니다.

설정값	설명
%q	HTTP 요청의 쿼리 값을 표시합니다.
%r	HTTP 요청의 Request line 전체를 표시합니다.
%s	응답에 사용된 HTTP 상태 코드를 표시합니다.
%S	http와 https를 구분하여 표시합니다.
%t	요청 처리를 마친 시간을 표시합니다.
%T	요청을 처리하는 데 소요된 시간을 표시합니다. (단위: 초)
%u	HTTP 인증에 사용된 사용자 이름을 표시합니다.
%U	HTTP 요청 URI를 표시합니다.
%v	호스트 헤더 필드 값을 표시합니다.

error_log/handlers

로그 메세지를 기록할 파일 및 콘솔에 대해 설정합니다.

구분	설명
자료형	object

error_log/handlers/file_handler

로그를 기록할 파일에 대해 설정합니다.

구분	설명
자료형	object

error_log/handlers/file_handler/file_name (필수 항목)

에러 로그를 저장할 파일의 경로를 설정합니다. 상대 경로는 /로 시작하지 않으면 자동으로 "$WEBTOB6_HOME_PATH/상대 경로"로 대체됩니다.

구분	설명
자료형	string
범위	255자 이내
기본값	"webtob_error.log"

파일 이름에 포함된 다음의 대체 문자열은 파일 생성 시 실제 값으로 변경됩니다. 예를 들어 FileName에 "log/error_%Y%%M%%D%.log"로 설정된 값은 로그 파일 생성 시 "$WEBTOB6_HOME_PATH/log/error_20231023.log"로 변환될 수 있습니다.

다음은 대체 문자열에 대한 설명입니다.

문자열	실제값
%Y%	년도(year)로 숫자 4개로 설정합니다. (예: 2009)
%M%	월(month)로 숫자 2개를 설정합니다. (예: 11)
%D%	일(day)로 숫자 2개를 설정합니다. (예: 05)
%h%	시(hour)로 숫자 2개를 설정합니다. (예: 10)
%m%	분(minute)으로 숫자 2개를 설정합니다. (예: 30)
%s%	초(second)로 숫자 2개를 설정합니다. (예: 45)

error_log/handlers/file_handler/rotate_by_seconds

새로운 로그 메시지 생성할 때 현재 존재하는 로그 파일이 지정된 시간보다 오래되면 새로운 로그 파일을 생성하도록 설정합니다.

구분	설명
자료형	integer
범위	0 ~ 86400
기본값	0

error_log/handlers/file_handler/valid_hours

새로운 로그 파일을 설정된 시간 단위로 생성하도록 설정합니다.

구분	설명
자료형	integer
범위	0 ~ 23
기본값	0

error_log/handlers/file_handler/rotate_by_file_size

새로운 로그 메시지 생성할 때 현재 존재하는 로그 파일의 크기가 지정된 크기보다 큰 경우 새로운 로그 파일을 생성하도록 설정합니다.

구분	설명
자료형	integer
단위	bytes
범위	0 ~ INT_MAX
기본값	0

error_log/handlers/file_handler/archive_file_name

새로운 로그 파일 생성 시점에 기존 로그 파일은 archive_file_name에 설정된 이름으로 변경됩니다. 이후 새로운 로그 파일은 file_name에 설정된 형식으로 생성됩니다.

구분	설명
자료형	string
범위	255자 이내

error_log/handlers/file_handler/enable_sync

로그가 기록될 때 파일에 즉시 기록할지 여부를 설정합니다.

구분	설명
자료형	boolean
기본값	false

다음은 설정값에 대한 설명입니다.

설정값	설명
true	로그 메시지들이 WebtoB 메모리에 버퍼링되지 않고 바로 파일에 기록됩니다. **[참고]** 로그를 바로 확인해야 하는 곳에서는 해당 옵션을 설정하면 문제가 발생하는 경우 쉽게 확인할 수 있습니다.

error_log/handlers/file_handler/permission

에러 로그 파일의 접근 권한을 설정합니다.

UNIX 계열 운영체제에서 사용하는 파일 접근 권한과 동일한 의미를 가지며, UNIX/Linux 환경에서만 사용할 수 있습니다.

구분	설명
자료형	string
범위	"0600" ~ "0777"

error_log/handlers/enable_console_handler

콘솔 창에 해당 로그를 출력할지 여부를 설정합니다.

구분	설명
자료형	boolean
기본값	false

2.8.2. 설정 예시

다음은 LOGGING 절을 설정한 예시입니다.

```
"logging": {
    "system_log": [
        {
            "name": "webtob",
            "level": "DEBUG",
            "handlers": {
                "file_handler": {
                    "file_name": "webtob-%Y%%M%%D%-%h%%m%%s%.log"
            }
        }
    ],
    "access_log": [
        {
            "name": "access_log1",
            "level": "INFO",
            "format": "DEFAULT",
            "handlers": {
                "file_handler": {
                    "file_name": "access-%Y%%M%%D%%s%.log"
                }
            }
        },
        {
            "name": "access_log2",
            "level": "INFO",
            "format": "DEFAULT",
            "handlers": {
                "file_handler": {
                    "file_name": "access2-%Y%%M%%D%%s%.log"
                }
            }
        }
    ],
    "error_log": {
        "level": "INFO",
        "format": "ERROR",
        "handlers": {
            "file_handler": {
                "file_name": "logs/errors%s%.log"
            }
        }
    }
}
```

2.9. NODE 절

NODE 절은 WebtoB 노드에 대한 구체적인 환경에 대해 설정합니다. NODE 절은 노드의 전체적인 동작과 관련된 설정을 정의합니다.

NODE 절에는 'WebtoB 시스템 경로', '기타 시스템의 전체적인 설정'들이 정의될 수 있습니다.

2.9.1. 설정 항목

다음은 NODE 절의 환경 설정 형식입니다.

```
"node": {
    "name": string
    "hth_count": integer,                        # 1 (1-255)
    #"worker_threads": integer,                  # 8 (1-100)
    #"hth_schedule": string,                     # "RR"
    #"connection_pool_size": integer,            # 8192 (1-INT_MAX)
    #"graceful_shutdown_timeout": integer        # 30 (1-INT_MAX)
    #"cache_key": string,                        # "HOST_URI"
    #"cache_entry": integer                      # 128 (0-INT_MAX)
    #"max_cache_memory_size": integer            # 100 (0-INT_MAX)
    #"cache_max_file_size": integer              # 8192 (0-INT_MAX)
    #"listen_backlog": integer                   # 4096 (0-INT_MAX)
    #"limit_request_body_size": integer          # 0 (0-INT_MAX)
    #"limit_request_header_field_count": integer # 100 (0-INT_MAX)
    #"limit_request_header_field_size": integer  # 8190 (0-INT_MAX)
    #"limit_request_line_size": integer          # 8190 (0-INT_MAX)
    #"system_filters": [string]
}
```

절과 설정 항목의 구성에 대한 기호나 내용에 대한 자세한 내용은 설정 항목 값의 형식 및 설정 방법을 참고합니다.

name (필수 항목)

노드의 이름을 설정합니다.

구분	설명
자료형	string
범위	127자 이내

다음은 name 설정 예시입니다. 프롬프트에 'user@webtob_node:~$'로 표시되는 경우 아래와 같이 name을 설정합니다.

```
"node": {
    "name": "webtob_node",
    ...
}
```

노드 이름은 호스트 이름과 같아야 합니다. 만약 호스트 이름과 다를 경우 실행할 수 없습니다.

hth_count

요청을 처리하는 HTH의 수를 설정합니다.

구분	설명
자료형	integer
범위	1 ~ 100
기본값	1

다음은 hth_count 설정 예시입니다. 아래와 같이 hth_count를 3으로 설정하면 'HTH-0', 'HTH-1', 'HTH-2'가 생성됩니다.

```
"node": {
    "name": "webtob_node",
    "hth_count": 3,
    ...
}
```

로그(webtob.log)를 통해 생성된 HTH를 확인할 수 있습니다.

```
EventManager(HTH-2) Thread Created. Thread ID: 4800
```

worker_threads

HTH에 할당할 작업 스레드의 수를 설정합니다. 작업 스레드는 정적 파일 처리, SSL/TLS 처리(Handshake, 암호화, 복호화), 압축 처리, HTTP 인증 처리를 하는 스레드입니다.

구분	설명
자료형	integer
범위	1 ~ 100
기본값	8

다음은 worker_threads 설정 예시입니다. 아래와 같이 worker_threads를 5로 설정하면 'HTMLS-0', 'HTMLS-1', 'HTMLS-2', 'HTMLS-3', 'HTMLS-4'가 생성됩니다.

```
"node": {
    "name": "webtob_node",
    "hth_count": 3,
    "worker_threads":5,
    ...
}
```

로그(webtob.log)를 통해 생성된 작업 스레드를 확인할 수 있습니다.

```
EventManager(HTMLS-4) Thread Created. Thread ID: 19271
```

hth_count × worker_threads의 수만큼 FD가 사용되기 때문에 hth_count와 worker_threads를 설정할 때 FD를 고려해야 합니다. 만약 설정된 값이 FD의 수를 초과하면 아래와 같은 에러 로그가 발생하며 서버가 종료됩니다.

```
Error: eventFd() failed: Too many open files
Too many open files
```

hth_schedule

HTH를 여러 개 설정했을 경우 요청을 처리할 HTH 지정 방법을 설정합니다.

구분	설명
자료형	string
범위	"RR"
기본값	"RR"

다음은 설정값에 대한 설명입니다.

설정값	설명
RR	Round Robin 방식으로 순차적으로 HTH에 요청을 할당합니다.

다음은 hth_schedule 설정 예시입니다.

```
"node": {
    "name": "webtob_node",
    "hth_count": 2,
    "worker_threads":8,
    "hth_schedule": "RR",
    ...
}
```

로그(webtob.log)를 통해 hth_count에 설정된 수만큼 순차적으로 요청이 할당되는 것을 확인할 수 있습니다.

```
[2024-07-25 06:49:19.113263413][DEBUG][HTH-
0][SVR_3455]WJP2RequestMessageHandler.cpp:87 serviceMessage: WJP Request Loop
Done.
```

```
[2024-07-25 06:49:19.113268083][DEBUG][HTH-0][NET_2004]EventManager.cpp:129 loop:
End of this loop.
[2024-07-25 06:49:19.425343033][DEBUG][HTH-1][NET_4000]EpollMultiplexer.cpp:33
select: Multiplexer awakened. Event Count=0.
[2024-07-25 06:49:19.425360647][DEBUG][HTH-1][NET_2004]EventManager.cpp:129 loop:
End of this loop.
[2024-07-25 06:49:20.114399067][DEBUG][HTH-0][NET_4000]EpollMultiplexer.cpp:33
select: Multiplexer awakened. Event Count=0.
[2024-07-25 06:49:20.114416261][DEBUG][HTH-0][NET_2004]EventManager.cpp:129 loop:
End of this loop.
[2024-07-25 06:49:20.426438758][DEBUG][HTH-1][NET_4000]EpollMultiplexer.cpp:33
select: Multiplexer awakened. Event Count=0.
[2024-07-25 06:49:20.426456148][DEBUG][HTH-1][NET_2004]EventManager.cpp:129 loop:
End of this loop.
```

connection_pool_size

HTH마다 커넥션을 관리하는 풀(Pool)의 크기를 설정합니다.

구분	설명
자료형	integer
범위	1 ~ INT_MAX
기본값	8192

hth_count, worker_threads와 마찬가지로 FD 수의 영향을 받기 때문에 설정 시 FD 수를 고려해야 합니다. 만약 설정된 값이 FD 수를 초과하면 아래와 같은 에러 로그가 발생하며 서버가 종료됩니다.

```
Error: eventFd() failed: Too many open files
Too many open files
```

graceful_shutdown_timeout

Graceful shutdown 명령 후 강제로 종료할 때까지 대기하는 타임아웃을 설정합니다.

구분	설명
자료형	integer
단위	초
범위	1 ~ INT_MAX

다음은 graceful_shutdown_timeout 설정 예시입니다.

```
{
```

```
  "node": {
    "name": "webtob_node",
    "hth_count": 1,
    "worker_threads":8,
    "graceful_shutdown_timeout":5
  },
  ...
```

cache_key

캐시된 데이터를 식별하기 위해 사용하는 키의 형식을 설정합니다.

구분	설명
자료형	string
범위	"HOST_URI" | "REAL_PATH"
기본값	"HOST_URI"

다음은 설정값에 대한 설명입니다.

설정값	설명
HOST_URI	요청 헤더의 HOST 값과 요청 URI를 사용하여 Key를 생성합니다. 동일한 REQUEST_URI라 하더라도 HTTP_HOST에 따라 VHOST가 달라질 수 있으며, 이는 실제 경로가 달라질 수 있음을 의미합니다. 그러나 VHOST가 다르더라도 실제 경로가 같다면 중복으로 캐싱되는 문제가 발생할 수 있습니다. 이 방식은 실제 경로를 계산하지 않으므로 REAL_PATH 방식보다 응답 속도가 빠릅니다. 따라서 VHOST별 DocRoot가 다르다면 HOST_URI 값을 사용하는 것이 좋습니다. JEUS에서 처리되거나 역방향 프록시로 처리되는 응답은 실제 경로를 계산할 수 없으므로 이 값을 사용합니다.
REAL_PATH	요청 URI의 실제 경로를 사용하여 Key를 생성합니다. 이 값은 SVRTYPE이 HTML인 경우에만 적용되고, HOST_URI를 사용할 때 발생할 수 있는 문제점을 보완합니다. HTTP_HOST가 다르더라도 동일한 요청 파일의 실제 경로가 같을 수 있다. 이 경우 HTTP_HOST에 상관없이 하나의 파일만 캐싱되므로 메모리 사용을 줄일 수 있습니다. 그러나 실제 경로를 계산해야 하기 때문에 HOST_URI를 사용하는 것보다 응답 속도가 다소 느릴 수 있습니다.

다음은 cache_key 설정 예시입니다. 이때 캐시를 저장하려면 destination 절에서 "enable_cache"를 true로 설정해야 합니다.

```
  "node": {
    "name": "webtob_node",
    "hth_count": 1,
    "worker_threads":8,
```

```
    "cache_key":"HOST_URI",
    "cache_entry":128,
    "max_cache_memory_size":100,
    "cache_max_file_size":8192
    ...
    "destination": {
      "htmls": [
      {
        "name": "htmls1",
        "enable_cache":true
      }
    ...
```

캐시 현황은 wsadmin의 cache-list에서 확인할 수 있습니다. 이때 cache_key 설정값이 "HOST_URI"인 경우 192.168.0.1:80/image.jpg, "REAL_PATH"인 경우 /home/tmax/webtob6/docs/image.jpg로 표시됩니다.

```
[wsadmin]>> cache-list
-------------------------------------------------------------------
| HTH-0: Cache List Info                                          |
-------------------------------------------------------------------
|          Cache key          |    Expired time     | Cache size |
-------------------------------------------------------------------
| 192.168.0.1:80/image.jpg | 2024-01-01 00:00:00 |         300|
-------------------------------------------------------------------
| Cache count : 1                                                 |
| Memory usages : 300                                             |
-------------------------------------------------------------------
```

cache_entry

HTH 캐시의 해시 테이블 Key의 크기를 설정합니다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	128

다음은 설정값에 대한 설명입니다.

설정값	설명
0	응답을 캐시하지 않습니다.

다음은 cache_entry 설정 예시입니다. 이때 캐시를 저장하려면 destination 절에서 "enable_cache"를 true로 설정해야 합니다.

```
"node": {
  "name": "webtob_node",
  "hth_count": 1,
  "worker_threads":8,
  "cache_key":"HOST_URI",
  "cache_entry":128,
  "max_cache_memory_size":100,
  "cache_max_file_size":8192
  ...
  "destination": {
    "htmls": [
    {
      "name": "htmls1",
      "enable_cache":true
    }
  ...
```

캐시 현황은 wsadmin의 cache-list에서 확인할 수 있습니다. 이때 cache_entry 설정값이 2인 경우 최대 2개의 행이 표시됩니다.

```
[wsadmin]>> cache-list
-------------------------------------------------------------------------------
-
| HTH-0: Cache List Info
|
-------------------------------------------------------------------------------
-
|                  Cache key                  |    Expired time     | Cache size
|
-------------------------------------------------------------------------------
-
| 192.168.0.1:80/image.jpg | 2024-01-01 00:00:24 |         345|
| 192.168.0.1:80/image1.jpg  | 2024-01-01 00:00:21 |         345|
-------------------------------------------------------------------------------
-
| Cache count : 2
|
| Memory usages : 1035
|
-------------------------------------------------------------------------------
-
```

max_cache_memory_size

HTH 프로세스가 캐시를 위해 사용하는 최대 메모리 사이즈를 설정합니다.

HTH별로 각각 캐시하기 때문에 해당 설정은 HTH 프로세스별로 각각 적용되는 값입니다. (모든 HTH 캐시가 사용하는 총 메모리 사이즈가 아님)

구분	설명
자료형	integer
단위	Mbytes
범위	0 ~ INT_MAX
기본값	100

다음은 설정값에 대한 설명입니다.

설정값	설명
0	응답을 캐시하지 않습니다.

다음은 max_cache_memory_size 설정 예시입니다. 이때 캐시를 저장하려면 destination 절에서 "enable_cache"를 true로 설정해야 합니다.

```
"node": {
  "name": "webtob_node",
  "hth_count": 1,
  "worker_threads":8,
  "cache_key":"HOST_URI",
  "cache_entry":128,
  "max_cache_memory_size":100,
  "cache_max_file_size":8192
  ...
  "destination": {
    "htmls": [
    {
      "name": "htmls1",
      "enable_cache":true
    }
  ...
```

캐시 현황은 wsadmin의 cache-list에서 확인할 수 있습니다. 이때 max_cache_memory_size 설정값이 100인 경우 cache로 저장된 이미지 크기는 100MB를 넘을 수 없습니다.

```
[wsadmin]>> cache-list
-----------------------------------------------------------------------------
-
| HTH-0: Cache List Info
|
-----------------------------------------------------------------------------
-
|                  Cache key                  |    Expired time     | Cache size
|
-----------------------------------------------------------------------------
-
| 192.168.0.1:80/image.jpg | 2024-01-01 00:00:24 |         345|
| 192.168.0.1:80/image1.jpg  | 2024-01-01 00:00:21 |         345|
-----------------------------------------------------------------------------
```

```
-
| Cache count : 2
|
| Memory usages : 1035
|
-----------------------------------------------------------------------------
-
```

cache_max_file_size

캐시할 수 있는 응답(헤더+본문) 하나의 최대 사이즈를 설정합니다.

구분	설명
자료형	integer
단위	bytes
범위	0 ~ INT_MAX
기본값	8192

다음은 설정값에 대한 설명입니다.

설정값	설명
0	응답을 캐시하지 않습니다.

다음은 cache_max_file_size 설정 예시입니다. 이때 캐시를 저장하려면 destination 절에서 "enable_cache"를 true로 설정해야 합니다.

```
"node": {
  "name": "webtob_node",
  "hth_count": 1,
  "worker_threads":8,
  "cache_key":"HOST_URI",
  "cache_entry":128,
  "max_cache_memory_size":100,
  "cache_max_file_size":8192
  ...
  "destination": {
    "htmls": [
    {
      "name": "htmls1",
      "enable_cache":true
    }
  ...
```

캐시 현황은 wsadmin의 cache-list에서 확인할 수 있습니다. 이때 cache_max_file_size 설정값이 8192인 경우 8192byte를 초과하는 이미지는 캐시되지 않습니다.

```
[wsadmin]>> cache-list
-------------------------------------------------------------------------------
-
| HTH-0: Cache List Info
|
-------------------------------------------------------------------------------
-
|                  Cache key                  |    Expired time     | Cache size
|
-------------------------------------------------------------------------------
-
| 192.168.0.1:80/image.jpg | 2024-01-01 00:00:24 |         345|
| 192.168.0.1:80/image1.jpg  | 2024-01-01 00:00:21 |         345|
-------------------------------------------------------------------------------
-
| Cache count : 2
|
| Memory usages : 1035
|
-------------------------------------------------------------------------------
-
```

listen_backlog

서비스 포트 연결 큐(listen backlog)의 크기를 설정합니다. 즉, 서비스 포트에서 동시에 처리할 수 있는 연결 시도 중인 소켓의 수를 제한하는 설정입니다.

구분	설명
자료형	integer
범위	1 ~ INT_MAX
기본값	4096

netstat 명령에서 SYN_RECVD 상태인 소켓의 수와 관련이 있을 수 있습니다.

다음은 listen_backlog 설정 예시입니다.

```
"node": {
  "name": "webtob_node",
  "hth_count": 1,
  "worker_threads":8,
  "listen_backlog":50
  ...
```

ss 명령어를 통해 변경된 Send-Q를 확인할 수 있습니다.

listen_backlog 설정값을 511 이상으로 설정할 경우 Send-Q 값은 511 이상으로 올라가지

않는데, 이는 "sysctl -w net.core.somaxconn=[설정값]"을 통해 시스템 값을 변경할 수 있습니다.

```
user@webtob_node:~/tmax/webtob6/$ ss -lnt | grep 80
State  Recv-Q    Send-Q             Local Address:Port    Peer Address:Port
LISTEN   0         50                  0.0.0.0:80           0.0.0.0:*
```

limit_request_body_size

사용자 요청에서 HTTP 요청 본문 크기를 제한할 경우 설정합니다.

HTTP 요청 본문이 2G(Long 타입) 이상인 경우는 JEUS 7 이후 버전부터 처리 가능합니다.

이 값은 HTTP 요청 헤더의 "Content-Length" 값을 기준으로 판단하며, Chunked-request의 경우 허용할 본문의 최대 크기를 설정합니다. 만약 설정된 값보다 큰 요청 본문이 들어올 경우 서버는 "413 Request Entity Too Large"로 응답합니다.

구분	설명
자료형	integer
단위	bytes
범위	0 ~ INT_MAX
기본값	0

다음은 설정값에 대한 설명입니다.

설정값	설명
0	요청 본문 크기를 제한하지 않습니다.

다음은 limit_request_body_size 설정 예시입니다.

```
"node": {
  "name": "webtob_node",
  "hth_count": 1,
  "worker_threads":8,
  "limit_request_body_size":100
  ...
```

limit_request_body_size를 100으로 설정했을 때, Content-Length가 100을 넘으면 아래와 같은 메시지가 출력됩니다.

```
* Closing connection 0
The body is too large.
```

limit_request_header_field_count

사용자 요청에서 HTTP 요청 헤더 필드 수를 제한할 경우 설정합니다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	100

다음은 설정값에 대한 설명입니다.

설정값	설명
0	헤더 필드의 수를 제한하지 않습니다.

다음은 limit_request_header_field_count 설정 예시입니다.

```
  "node": {
    "name": "webtob_node",
    "hth_count": 1,
    "worker_threads":8,
    "limit_request_header_field_count":1
    ...
```

limit_request_header_field_count를 1로 설정했을 때, 헤더 필드 수가 1을 넘으면 아래와 같은 메시지가 출력됩니다.

```
* Closing connection 0
The number of request header fields exceeds the server limit.
```

limit_request_header_field_size

사용자 요청에서 HTTP 요청 헤더 필드 각각의 크기를 제한할 경우 설정합니다.

구분	설명
자료형	integer
단위	bytes
범위	0 ~ 16382
기본값	8190

다음은 설정값에 대한 설명입니다.

설정값	설명
0	헤더 필드의 크기를 제한하지 않습니다.

10,000bytes를 초과하는 요청 헤더를 사용하려면 "WJPv2"를 이용해야 합니다.

다음은 limit_request_header_field_size 설정 예시입니다.

```
"node": {
  "name": "webtob_node",
  "hth_count": 1,
  "worker_threads":8,
  "limit_request_header_field_size":100
  ...
```

limit_request_header_field_size를 100으로 설정했을 때, 헤더 필드 각각의 크기가 100을 넘으면 아래와 같은 메시지가 출력됩니다.

```
* Closing connection 0
The size of a request header field exceeds the server limit.
```

limit_request_line_size

사용자 요청에서 HTTP 요청 라인 크기를 제한할 경우 설정합니다.

구분	설명
자료형	integer
단위	bytes
범위	0 ~ 16382
기본값	8190

다음은 설정값에 대한 설명입니다.

설정값	설명
0	요청 라인의 길이를 제한하지 않습니다.

10,000bytes를 초과하는 요청 라인을 사용하려면 "WJPv2"를 이용해야 합니다.

다음은 limit_request_line_size 설정 예시입니다.

```
"node": {
  "name": "webtob_node",
  "hth_count": 1,
  "worker_threads":8,
  "limit_request_line_size":100
```

```
    ...
```

limit_request_line_size를 100으로 설정했을 때, 요청 라인의 길이가 100을 넘으면 아래와 같은 메시지가 출력됩니다.

```
* Closing connection 0
Request Uri too Wrong
```

system_filters

시스템 이벤트(ON_START, ON_STOP) 발생 시 적용할 필터를 설정합니다.

구분	설명
자료형	array(string)
범위	15개 이내(255자 이내)

2.9.2. 설정 예시

다음은 NODE 절을 설정한 예시입니다.

```
"node": {
    "name": "webtob_node",
    "hth_count": 1,
    "worker_threads": 8,
    "hth_schedule": "RR",
    "connection_pool_size": 8192,
    "graceful_shutdown_timeout": 30,
    "listen_backlog": 4096,
    "cache_key": "HOST_URI",
    "cache_entry": 128,
    "max_cache_memory_size": 100,
    "cache_max_file_size": 8192,
    "limit_request_body_size": 0,
    "limit_request_header_field_count": 100,
    "limit_request_header_field_size": 8190,
    "limit_request_line_size": 8190,
    "system_filters" : [ filter1 ]
}
```

2.10. SERVER 절

SERVER 절에서는 WebtoB가 클라이언트의 요청을 받을 수 있는 접속 정보(Port, SSL 등)를 설정합니다.

기본적으로 WebtoB에서 요청을 처리할 수 있는 프로토콜마다 SERVER 절에 설정해야 합니다.

WebtoB에서 제공하는 SERVER 타입으로는 **'http'**, **'wjp'**, **'admin'**이 있으며, 각각 타입에 대해서 세부적으로

설정할 수 있습니다.

2.10.1. 설정 항목

다음은 SERVER 절의 환경 설정 형식입니다.

```
"server": {
    "http": {
        #"common_config": {                             # COMMON
            #"doc_root": string,                        # "docs/", $ENV, R.PATH
            #"error_document": [string],
            #"enable_keepalive": boolean,               # true
            #"keepalive_timeout": integer,              # 60 (1-3600)
            #"keepalive_max": integer                   # 0 (0-INT_MAX)
            #"keepalive_error_status_code": [integer]
            #"service_order": string,                   # "uri,ext"
            #"access_log": string
            #"headers": [string]
            #"enable_dos_block": boolean                # false
            #"dos_block_table_size": integer            # 3097 (1-INT_MAX)
            #"dos_block_page_count": integer            # 5 (0-INT_MAX)
            #"dos_block_page_interval": integer         # 1 (1-INT_MAX)
            #"dos_block_site_count": integer            # 50 (0-INT_MAX)
            #"dos_block_site_interval": integer         # 1 (1-INT_MAX)
            #"dos_block_period": integer                # 30 (1-INT_MAX)
            #"dos_block_white_list": [string]
            #"url_rewrite_config": string
            #"alias": [string]
            #"index_name": string                       # "index.html"
            #"filters": [string]
            #"rpaf_header": string
            #"enable_directory_index": boolean          # true
        },
        "http_servers": [
            {
                #"common_config": {...},                # COMMON
                "name": string,
                #"port": integer,                       # 80 | 443 (1-65535)
                #"enable_ssl": boolean,                 # false
                #"ssl_name": [string],
                #"idle_timeout": integer                # 300 (1-3600)
                #"initial_connection_timeout": integer  # 10 (0-INT_MAX)
                #"request_header_timeout": integer      # 60 (0-INT_MAX)
                #"request_body_timeout": integer        # 0 (0-INT_MAX)
                #"idle_timeout_status": integer         # 500 (511-599)
                #"vhosts": [
                    {
                        #"common_config": {...},        # COMMON
                        "name": string,
                        "host_name": [string]
                    }
                ]
            }
        ],
        #"mime_type_file": string,                      # "mime.types"
        #"default_mime_type": string                    # "text/html"
```

```
    },
    #"wjp": {
        "wjp_servers": [
            {
                "name": string,
                #"min_proc": integer,                   # 1 (1-INT_MAX)
                #"max_proc": integer,
                #"svr_chk_time": integer                # 60 (0-3600)
                #"flow_control": integer                # 50 (1-INT_MAX)
                #"max_jengine_count": integer           # 64 (1-INT_MAX)
                #"ping_timeout_status": integer         # 503 (511-599)
            }
        ],
        #"port": integer,                               # 9900 (1-65535)
        #"enable_ssl": boolean,                         # false
        #"ssl_name": string
    }

    #"admin": {
        #"port": integer                                # 9090 (1-65535)
        #"access_log": string
    }

    #"tcp": {
        "tcp_servers": [
            {
                "name": string
                "port": integer                         # (1-65535)
                "server_address": [string]
                #"idle_timeout": integer                # 300 (0-INT_MAX)
                #"initial_connection_timeout"           # 10 (0-INT_MAX)
            }
        ]
    }
```

절과 설정 항목의 구성에 대한 기호나 내용에 대한 자세한 내용은 설정 항목 값의 형식 및 설정 방법을 참고합니다.

http (필수 항목)

HTTP 요청을 처리하기 위한 접속 정보를 설정합니다.

구분	설명
자료형	object

http/common_config

HTTP 서버의 공통적인 설정입니다. 상위 항목에 설정하면 하위 항목에는 설정하지 않아도 상위 항목의 설정이 반영됩니다.

구분	설명
자료형	object
설정 우선순위	설정할 때 적용되는 우선순위는 다음과 같습니다. 1. vhosts 2. http_servers 3. http

http/common_config/doc_root

서비스할 문서가 위치한 최상위 경로를 설정합니다.

환경 변수를 사용할 수 있으며, 상대 경로로 설정한 경우 $WEBTOB6_HOME_PATH를 기준으로 절대 경로로 변환하여 설정됩니다.

구분	설명
자료형	string
범위	255자 이내
기본값	"docs"

다음은 doc_root 설정 예시입니다.

```
  "server": {
    "http": {
      "common_config": {
        "doc_root": "docs",
}
```

common_config와 이후에 나올 http_servers에서 모두 doc_root를 설정한 경우 http_servers > common_config 순으로 http_servers 설정이 우선 적용됩니다.

http/common_config/error_document

HTTP 에러 페이지를 사용자가 지정한 페이지로 대신 사용할 경우 ERRORDOCUMENT 절에 정의한 이름을 설정합니다.

구분	설명
자료형	array(string)
범위	64개 이내(255자 이내)

다음은 error_document 설정 예시입니다. 이때 error_document 목록에서 설정할 error_document의 이름을 문자열 형식으로 작성하고, 설정할 error_document는 error_document 절에 명시되어 있어야 합니다.

```
"server": {
  "http": {
    "common_config": {
      "error_document": ["error1"]
...
"error_document": {
  "error_document_list": [
    {
      "name": "error1",
      "status": 404,
      "url": "/htmls/not_found.html"
    }
  ]
},
```

http/common_config/enable_keepalive

Keepalive(HTTP persistent connection) 사용 여부를 설정합니다.

구분	설명
자료형	boolean
기본값	true

다음은 설정값에 대한 설명입니다.

설정값	설명
true	사용자는 연결을 재사용하여 한 번의 연결로 여러 개의 요청을 처리할 수 있습니다.
false	사용자는 요청을 처리하기 위해 매번 소켓을 다시 연결해야 합니다.

다음은 enable_keepalive 설정 예시입니다. keepalive_timeout과 keepalive_max를 설정하기 위해서는 enable_keepalive를 true 설정해야 합니다.

```
"server": {
  "http": {
    "common_config": {
      "enable_keepalive": true,
      "keepalive_timeout": 60,
      "keepalive_max":0,
    },
```

http/common_config/keepalive_timeout

Keepalive를 유지하는 시간을 설정합니다. 사용자 요청 처리가 끝난 후 설정된 시간이 지나면 연결을 끊습니다.

구분	설명
자료형	integer

구분	설명
단위	초
범위	1 ~ 3600
기본값	60

다음은 keepalive_timeout 설정 예시입니다.

```
"server": {
  "http": {
    "common_config": {
      "enable_keepalive": true,
      "keepalive_timeout": 10,
      "keepalive_max":0,
    },
```

wsadmin의 ci에서 Idle_time을 통해 연결 시간을 확인할 수 있습니다. 브라우저에서 호출할 경우 각 브라우저마다 연결 시간 정책이 있기 때문에 keepalive_timeout 설정값보다 먼저 연결이 끊길 수 있습니다.

```
[wsadmin]>> ci
-------------------------------------------------------------------------------
-----------------------------------------------
| HTH-0 : Connection info
|
-------------------------------------------------------------------------------
-----------------------------------------------
| No | Server |    Local Address    |   Remote Address    | Remote Type | Ssl |
Status | Request Count | Idle Time | Mapping No |
-------------------------------------------------------------------------------
-----------------------------------------------
| 5  | http1  | 192.168.0.1:80 | 192.168.0.51:50965 |   CLIENT    | No  | READY
|       0       |     7     |     -1     |
-------------------------------------------------------------------------------
-----------------------------------------------
```

http/common_config/keepalive_max

Keepalive 요청 건수를 제한할 경우 설정합니다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	0

다음은 설정값에 대한 설명입니다.

설정값	설명
0	요청 건수를 제한하지 않습니다.

다음은 keepalive_max 설정 예시입니다.

```
"server": {
  "http": {
    "common_config": {
      "enable_keepalive": true,
      "keepalive_timeout": 60,
      "keepalive_max":2,
    },
```

wsadmin의 ci에서 Request Count를 통해 연결 횟수을 확인할 수 있습니다.

```
[wsadmin]>> ci
-------------------------------------------------------------------------------
-----------------------------------------------
| HTH-0 : Connection info
|
-------------------------------------------------------------------------------
-----------------------------------------------
| No | Server |    Local Address    |   Remote Address    | Remote Type | Ssl |
Status | Request Count | Idle Time | Mapping No |
-------------------------------------------------------------------------------
-----------------------------------------------
| 5  | http1  | 192.168.0.1:80 | 192.168.0.51:50965 |   CLIENT    | No  | READY
|       0       |     7     |     -1     |
-------------------------------------------------------------------------------
-----------------------------------------------
```

http/common_config/keepalive_error_status_code

에러 응답을 보낸 이후 사용자 연결을 유지(persistent connection, keep-alive)하는 경우 해당되는 HTTP 상태 코드를 설정합니다.

구분	설명
자료형	array(integer)
범위	64개 이내(300 - 303 | 305 - 599)

WebtoB가 사용자 요청에 대해 "304 Not Modified"를 제외한 3xx나 4xx 또는 5xx HTTP 상태 코드로 응답할 경우 연결을 유지하지 않고 응답을 보낸 이후 연결을 종료합니다.

다음은 "302 Found"와 "404 Not Found" 응답을 보낸 후 WebtoB가 클라이언트의 연결을 종료하지 않고 유지하는 예시입니다.

```
...
"server": {
  "http": {
    "common_config": {
      "enable_keepalive": true,
      "keepalive_timeout": 60,
      "keepalive_max":0,
      "keepalive_error_status_code": [302, 404],
    },
...
```

http/common_config/service_order

사용자 요청을 처리할 Destination을 결정할 때 SERVICE 절의 URI, EXT 간 우선순위를 설정합니다.

구분	설명
자료형	string
범위	"uri,ext" | "ext,uri"
기본값	"uri,ext"

다음은 설정값에 대한 설명입니다.

설정값	설명
uri,ext	URI 설정을 먼저 확인하고, 해당되는 설정이 없는 경우 EXT의 설정을 확인합니다.
ext,uri	EXT 설정을 먼저 확인하고, 해당되는 설정이 없는 경우 URI의 설정을 확인합니다.

URI 설정과 EXT 설정 모두 해당되지 않을 경우 기본 HTML 서비스(Worker Thread)에서 해당 요청을 처리합니다.

다음은 service_order 설정 예시입니다.

```
...
"server": {
  "http": {
    "common_config": {
      "service_order: : "uri,ext"
    },
...
```

http/common_config/access_log

액세스 로그에 해당되는 LOGGING 절 이름을 설정합니다.

구분	설명
자료형	string
범위	255자 이내

다음은 access_log 설정 예시입니다. 설정값으로 지정한 access_log는 logging 절의 access_log 항목에 정의되어 있어야 합니다.

```
...
"server": {
  "http": {
    "common_config": {
      "access_log": "access_log1"
    },
    ...
"logging": {
  "access_log": [
    {
      "name": "access_log1",
      "level": "INFO",
      "format": "DEFAULT",
      ...
...
```

http/common_config/headers

적용할 HEADERS 절 이름을 설정합니다.

구분	설명
자료형	array(string)
범위	15개 이내(255자 이내)

다음은 headers 설정 예시입니다. headers 절에 있는 headers_list에서 적용할 헤더의 이름을 문자열 형식으로 설정합니다. 이때 여러 개의 헤더를 설정할 수 있습니다.

```
...
"server": {
  "http": {
    "common_config": {
      "headers": ["header1","header2"]
    },
...
"headers": {
  "headers_list": [
    {
      "name": "header1",
      "action": "AppendResponse",
      "field_name": "Content-Type",
      "field_value": "; charset=ISO"
    },
    {
```

```
        "name": "header2",
        "action":"AddRequest",
        "field_name":"Accept-Encoding",
        "field_value":"GZIP"
      },
    ]
  }
```

http/common_config/enable_dos_block

DoS 공격 차단 기능의 사용 여부를 설정합니다.

구분	설명
자료형	boolean
기본값	false

다음은 enable_dos_block 설정 예시입니다. DoS 공격 차단 기능을 사용하려면 true로 설정합니다.

```
...
"server": {
  "http": {
    "common_config": {
        ...
        "enable_dos_block":true,
        "dos_block_table_size":3097,
        "dos_block_site_count":50,
        "dos_block_site_interval":1,
        "dos_block_period":30
        ...
    },
...
```

http/common_config/dos_block_table_size

DoS 공격 차단 기능에서 DoS 공격으로 판단할 IP 주소의 최대 개수를 설정합니다. 이때 설정값보다 초과하여 들어오는 IP 주소는 모두 차단합니다.

구분	설명
자료형	integer
범위	1 ~ INT_MAX
기본값	3097

테이블은 HTH별로 관리되므로, HTH 값을 1보다 크게 설정하면 관리하는 IP 주소의 수는 "dos_block_table_size × hth_count"가 된다.

다음은 dos_block_table_size 설정 예시입니다. 들어오는 IP는 개별적으로 관리되지 않고, IP별로 객체로 관리됩니다. 예를 들어 192.168.0.1과 192.168.0.2가 들어오면 각각의 IP가 별도의 객체로 관리됩니다.

```
...
"server": {
  "http": {
    "common_config": {
        ...
        "enable_dos_block":true,
        "dos_block_table_size":3097,
        "dos_block_site_count":50,
        "dos_block_site_interval":1,
        "dos_block_period":30
        ...
    },
...
```

http/common_config/dos_block_page_count

같은 페이지에 대한 요청 횟수를 설정합니다.

dos_block_page_interval 설정 시간 동안 같은 페이지를 설정된 횟수 이상 요청한 사용자 IP 주소를 dos_block_period 동안 차단합니다.

구분	설명
자료형	integer
범위	1 ~ INT_MAX
기본값	5

다음은 설정값에 대한 설명입니다.

설정값	설명
0	같은 페이지에 대한 DoS 공격을 체크하지 않습니다.

다음은 dos_block_page_count 설정 예시입니다. interval 내에 count(5개)보다 많은 요청이 발생하면 해당 IP 주소를 차단합니다. dos_block_page_count는 dos_block_page_interval과 함께 사용됩니다.

```
...
"server": {
  "http": {
    "common_config": {
        ...
        "enable_dos_block":true,
        "dos_block_table_size":1,
        "dos_block_page_count":5,
        "dos_block_page_interval":1,
        "dos_block_period":30,
        ...
    },
```

```
...
```

http/common_config/dos_block_page_interval

같은 페이지에 대한 DoS 공격을 판단하는 주기를 설정합니다.

구분	설명
자료형	integer
단위	초
범위	1 ~ INT_MAX
기본값	1

다음은 dos_block_page_interval 설정 예시입니다. interval(100초) 내에 count보다 많은 요청이 발생하면 해당 IP 주소를 차단합니다. dos_block_page_interval은 dos_block_page_count와 함께 사용됩니다.

```
...
"server": {
  "http": {
    "common_config": {
        ...
        "enable_dos_block":true,
        "dos_block_table_size":1,
        "dos_block_page_count":5,
        "dos_block_page_interval":1,
        "dos_block_period":30,
        ...
    },
...
```

http/common_config/dos_block_site_count

사이트 전체에 대한 요청 횟수를 설정합니다.

dos_block_site_interval 설정 시간 동안 사이트의 모든 페이지를 설정된 횟수 이상 요청한 사용자 IP 주소를 dos_block_period 동안 차단합니다.

구분	설명
자료형	integer
범위	1 ~ INT_MAX
기본값	50

다음은 설정값에 대한 설명입니다.

설정값	설명
0	사이트 요청에 대한 DoS 공격을 체크하지 않습니다.

http/common_config/dos_block_site_interval

같은 사이트에 대한 DoS 공격을 판단하는 주기를 설정합니다.

구분	설명
자료형	integer
단위	초
범위	1 ~ INT_MAX
기본값	1

http/common_config/dos_block_period

특정 사용자 IP 주소가 차단되는 기간을 설정합니다.

구분	설명
자료형	integer
단위	초
범위	1 ~ INT_MAX
기본값	30

http/common_config/dos_block_white_list

DoS 공격 차단에서 제외한 IP 주소를 설정합니다. 앞단에 프록시 서버를 사용할 경우 해당 서버의 IP 주소를 설정합니다.

구분	설명
자료형	array(string)
범위	256개 이내(255자 이내)
기본값	30

다음은 dos_block 설정 예시입니다. 이 경우 table_size(1)의 IP 주소가 interval(100초) 동안 count(3개)보다 많은 요청이 발생하면 해당 IP 주소를 period(100초)동안 차단합니다. 단, white_list에 있는 IP 주소는 DoS 차단 관리 IP 주소에서 제외합니다.

```
...
"server": {
  "http": {
    "common_config": {
        ...
```

```
        "enable_dos_block":true,
        "dos_block_table_size":1,
        "dos_block_site_count":3,
        "dos_block_site_interval":100,
        "dos_block_period":100,
        "dos_block_white_list":["192.168.0.1"],
        ...
    },
...
```

http/common_config/url_rewrite_config

URL 재작성 기능을 사용하기 위한 설정 파일의 경로를 설정합니다.

구분	설명
자료형	string
범위	255자 이내

다음은 url_rewrite_config 설정 예시입니다. 해당 경로에 정규식으로 작성된 설정 파일을 위치시킵니다.

```
...
"server": {
  "http": {
    "common_config": {
        ...
        "url_rewrite_config":"/home/tmax/webtob6/config/rewrite9.conf",
        ...
    },
...
```

http/common_config/alias

특정 요청의 URI를 realpath로 변경하려면 ALIAS 절에 정의한 이름을 설정합니다.

구분	설명
자료형	array(string)
범위	64개 이내(255자 이내)

다음은 alias 설정 예시입니다. 이 경우 url(/external/path/)을 호출했을 때 real_path(/home/tmax/webtob6/docs/)로 호출됩니다.

```
...
"server": {
  "http": {
    "common_config": {
        ...
          "alias": {
```

```
            "alias_list":[
              {
                "name":"alias1",
                "url":"/external/path/",
                "real_path":"/home/tmax/webtob6/docs/"
              }
            ]
          }
        ...
      },
  ...
```

http/common_config/index_name

디렉터리에 대한 요청을 하는 경우 기본으로 처리할 문서의 이름을 설정합니다. 예를 들어 클라이언트가 보낸 요청이 "/somedir/"이고, 해당 경로가 디렉터리인 경우 "/somedir/index.html"을 서비스합니다.

구분	설명
자료형	string
범위	255자 이내
기본값	"index.html"

http/common_config/filters

요청을 처리할 때 아래와 같은 이벤트 발생 시 적용할 필터를 설정합니다.

```
AFTER_SEND_REQUEST, AFTER_SEND_RESPONSE, BEFORE_SEND_REQUEST, BEFORE_SEND_RESPONSE, RECEIVE_REQUEST,
RECEIVE_RESPONSE
```

구분	설명
자료형	array(string)
범위	15개 이내(255자 이내)

http/common_config/rpaf_header

프록시 등을 거치면서 변경된 원격 IP 주소의 값을 원래 요청이 발생된 호스트 값으로 설정합니다.

구분	설명
자료형	string
범위	255자 이내

rpaf_header 추가를 통해 사용자가 헤더 지정이 가능하고, rpaf_header가 설정된 헤더가 들어올 경우 헤더의 IP 주소 값으로 원격 IP를 변경합니다.

access.log에서 설정된 원격 IP 주소를 확인할 수 있습니다.

```
"rpaf_header" = "X-Forwarded-For"
```

http/common_config/enable_directory_index

디렉터리에 대한 요청을 처리할 때 index_name 파일이 존재하지 않을 경우 Directory Index 기능의 사용 여부를 설정합니다.

구분	설명
자료형	boolean
기본값	true

http/http_servers (필수 항목)

HTTP를 처리할 서버들의 접속 정보를 설정합니다.

구분	설명
자료형	array(object)
범위	100개 이내

http/http_servers/common_config

HTTP 서버의 공통적인 설정입니다. 상위 항목에 설정하면 하위 항목에는 설정하지 않아도 상위 항목의 설정이 반영됩니다.

구분	설명
자료형	object
설정 우선순위	설정할 때 적용되는 우선순위는 다음과 같습니다. 1. vhosts 2. http_servers 3. http

http/http_servers/name (필수 항목)

HTTP 서버의 이름을 설정합니다.

구분	설명
자료형	string

구분	설명
범위	31자 이내

http/http_servers/port

사용자가 접속할 수 있는 HTTP 서버의 서비스 포트를 설정합니다.

구분	설명
자료형	integer
범위	1 ~ 65535
기본값	80 또는 443 (SSL을 사용할 경우에는 443을 기본값으로 사용)

http/http_servers/enable_ssl

HTTP 연결을 위한 포트를 SSL/TLS 프로토콜을 사용하여 서비스할지 여부를 설정합니다.

구분	설명
자료형	boolean
기본값	false

http/http_servers/ssl_name

적용할 SSL 절 항목을 설정합니다. 단, enable_ssl을 true로 설정한 경우 적용됩니다.

구분	설명
자료형	array(string)
범위	64개 이내(31자 이내)

다음은 ssl_name 설정 예시입니다.

```
...
"server": {
  "http": {
    "http_servers": {
        ...
        "enable_ssl": true,
        "ssl_name": ["ssl1"]
        ...
    },
...
```

http/http_servers/idle_timeout

사용자가 연결한 소켓에서 데이터를 읽거나 쓸 때 적용되는 타임아웃을 설정합니다.

사용자 요청을 처리할 때 설정된 시간 동안 사용자가 소켓에 데이터를 쓰지 않거나 소켓으로부터 데이터를 읽지 않으면 소켓을 닫습니다.

구분	설명
자료형	integer
단위	초
범위	1 ~ 3600
기본값	300

http/http_servers/initial_connection_timeout

사용자로부터 첫 번째 요청을 기다리기 위한 타임아웃을 설정합니다.

TCP 연결을 맺은 후 설정된 시간 동안 사용자가 어떠한 요청도 보내지 않으면 해당 소켓을 닫습니다. 해당 설정은 SSL 연결 시에도 적용됩니다. 사용자가 설정된 시간 내에 SSL 연결을 완료하지 않거나 완료 후에도 요청을 보내지 않으면 소켓을 닫습니다.

구분	설명
자료형	integer
단위	초
범위	0 ~ INT_MAX
기본값	10

다음은 설정값에 대한 설명입니다.

설정값	설명
0	첫 번째 요청을 기다리는 시간이 제한되지 않습니다.

http/http_servers/request_header_timeout

HTTP 요청 본문이 있는 경우 본문을 기다리기 위한 타임아웃을 설정합니다.

사용자가 HTTP 요청 본문을 보내는 동안 적용되는 설정으로, 설정된 시간 내에 요청 본문을 모두 보내지 않으면 "408 Request Timeout"으로 응답합니다.

구분	설명
자료형	integer
단위	초
범위	0 ~ INT_MAX

구분	설명
기본값	60

다음은 설정값에 대한 설명입니다.

설정값	설명
0	사용자가 HTTP 요청 본문을 보내는 시간을 제한하지 않습니다.

http/http_servers/idle_timeout_status

유휴(Idle) 타임아웃이 발생하면 해당 설정 값으로 HTTP 상태 코드를 설정해 응답합니다.

구분	설명
자료형	integer
범위	511 ~ 599
기본값	500

http/http_servers/vhosts

HTTP 서버가 호출 주소에 따라 다른 서비스를 제공하는 가상 호스트 기능을 설정합니다.

구분	설명
자료형	array(object)
범위	64개 이내

다음은 vhosts 설정 예시입니다.

```
...
"server": {
  "http": {
    "http_servers": {
        ...
        "vhosts":[
          {
            "name":"vhost1",
            "host_name":["www.vh1.com"],
            "common_config":{...}
          },
          {
            "name":"vhost2",
            "host_name":["www.vh2.com"],
            "common_config":{...}
          }
        ],
    }
  }
  ...
```

```
  },
  ...
```

http/http_servers/vhosts/name (필수 항목)

가상 호스트의 이름을 설정합니다.

구분	설명
자료형	string
범위	64개 이내

http/http_servers/vhosts/host_name (필수 항목)

가상 호스트에 접근할 때 사용자가 사용할 호스트 이름을 설정합니다. 이때 각 호스트 이름을 다르게 설정하여 가상 호스트를 구분합니다.

구분	설명
자료형	array(string)
범위	10개 이내(127자 이내)

http/mime_type_file

MIME-Type과 확장자를 매핑하는 MIME-Type 설정 파일의 경로를 설정합니다.

이 경로는 절대 경로나 $WEBTOB6_HOME_PATH의 상대 경로로 지정할 수 있습니다. 아무 값도 설정하지 않으면 해당 기능이 사용되지 않습니다.

구분	설명
자료형	string
범위	255자 이내
기본값	"mime.types"

http/default_mime_type

MIME-Type을 결정할 수 없는 문서에 대한 기본 Content-Type을 설정합니다.

구분	설명
자료형	string
범위	127자 이내
기본값	"text/html"

다음은 mime_type 설정 예시입니다.

```
...
"server": {
  "http": {
    "http_servers": {
        ...
          "mime_type_file":"mime.types",
          "default_mime_type":"text/html"
        ...
    },
...
```

우선순위가 mime_type보다 ext가 더 높기 때문에 mime_type을 설정하려면 ext 설정을 제거해야 합니다.

wjp

JEUS와 통신을 위한 WJP의 접속 정보를 설정합니다.

구분	설명
자료형	object

wjp/port

WebtoB와 JEUS를 연동할 때 필요한 서비스 포트를 설정합니다.

구분	설명
자료형	integer
범위	1 ~ 65535
기본값	9900

WebtoB가 해당 포트를 열면 JEUS가 "domain.xml"에 다음과 같은 설정을 통하여 연결합니다.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<domain version="8.0" xmlns="http://www.tmaxsoft.com/xml/ns/jeus">
    ...
    <servers>
        <server>
            <web-engine>
                <web-connections>
                    <webtob-connector>
                        <name>webtob1</name>
                        <wjp-version>2</wjp-version>
                        <registration-id>MyGroup</registration-id>
                        <network-address>
```

```
                        <port>9900</port>
                    </network-address>
                    <thread-pool>
                        <number>10</number>
                    </thread-pool>
                </webtob-connector>
            </web-connections>
        </web-engine>
    ...
```

wjp/enable_ssl

JEUS 연결을 위한 포트를 SSL/TLS 프로토콜을 사용하여 서비스할지 여부를 설정합니다.

구분	설명
자료형	boolean
기본값	false

WebtoB SSL을 통해 생성한 Truststore을 JEUS에 설정하여 연결할 수 있습니다.

wjp/ssl_name

적용할 SSL 절 항목을 설정합니다. 단, enable_ssl을 true로 설정한 경우 적용됩니다.

구분	설명
자료형	string
범위	31자 이내

wjp/wjp_servers (필수 항목)

WJP를 처리할 서버들의 접속 정보를 설정합니다.

구분	설명
자료형	array(object)
범위	100개 이내

다음은 wjp_servers 설정 예시입니다.

```
...
"server": {
  "wjp": {
    "wjp_servers": [
      {
```

```
          "name": "JeusServer1",
          "svr_chk_time": 60
        },
        {
          "name": "JeusServer2",
          "svr_chk_time": 30
        }
      ]

  ...
```

wjp/wjp_servers/name (필수 항목)

WJP 서버의 이름을 설정합니다.

구분	설명
자료형	string
범위	15자 이내

wjp/wjp_servers/svr_chk_time

JEUS와의 연결이 정상인지 확인하기 위해 체크하는 시간을 설정합니다.

서비스 요청이 없는 준비 상태의 연결에 대해서 연속된 2회의 svr_chk_time에 의한 KeepAlive 요청에도 응답이 없으면 해당 연결에 이상이 발생했다고 인식하고 해당 연결을 단절하여 서비스 분배에서 제외합니다.

구분	설명
자료형	integer
단위	초
범위	1 ~ 3600
기본값	60

wjp/wjp_servers/flow_control

서버로부터 받는 응답량을 제어하기 위해 응답 버퍼의 최대 크기를 설정합니다. 이때 설정값이 작을수록 페이지 호출이 느려지고, 클수록 빨라집니다.

구분	설명
자료형	integer
범위	1 ~ INT_MAX
기본값	50

wjp/wjp_servers/max_jengine_count

하나의 서버에 연결될 수 있는 최대 JEUS 서버 수를 설정합니다.

구분	설명
자료형	integer
범위	1 ~ INT_MAX
기본값	64

wjp/wjp_servers/ping_timeout_status

JSV 서버로 PING을 보내고 응답을 기다리던 중 타임아웃이 발생했을 때 응답할 HTTP 상태 코드를 설정합니다.

구분	설명
자료형	integer
범위	511 ~ 599
기본값	503

admin

WebtoB Admin 서버의 접속 정보를 설정합니다. Admin 서버에서 제공하는 WebtoB API를 호출할 때 사용합니다.

구분	설명
자료형	object

admin/port

Admin 서버의 포트 번호를 설정합니다.

구분	설명
자료형	integer
범위	1 ~ 65535
기본값	9090

설정한 포트 번호는 wsadmin 실행 시 로그에서 확인할 수 있습니다.

admin/access_log

액세스 로그에 해당되는 LOGGING 절 이름을 설정합니다.

구분	설명
자료형	string
범위	255자 이내

tcp

TCP의 접속 정보를 설정합니다. TCP 서버는 특정 포트나 IP 주소로 들어오는 TCP 연결을 다른 서버로 전송하는 중계(proxy) 역할을 합니다. 이때 TCP 서버는 전송되는 데이터를 전혀 해석하지 않습니다.

구분	설명
자료형	object

tcp/tcp_servers (필수 항목)

TCP 요청을 처리하기 위한 서버들의 접속 정보를 설정합니다.

구분	설명
자료형	array(object)
범위	100개 이내

tcp/tcp_servers/name (필수 항목)

TCP 서버의 이름을 설정합니다.

구분	설명
자료형	string
범위	31자 이내

tcp/tcp_servers/port (필수 항목)

사용자가 접속할 수 있는 TCP 서버의 서비스 포트를 설정합니다.

구분	설명
자료형	integer
범위	1 ~ 65535

tcp/tcp_servers/server_address (필수 항목)

클라이언트의 요청을 처리할 서버들의 IP 주소와 포트 번호를 설정합니다.

구분	설명
자료형	array(string)
범위	100개 이내(31자 이내)

tcp/tcp_servers/idle_timeout

사용자가 연결한 소켓에서 데이터를 읽거나 쓸 때 적용되는 타임아웃을 설정합니다.

사용자 요청을 처리할 때 설정된 시간 동안 사용자가 소켓에 데이터를 쓰지 않거나 소켓으로부터 데이터를 읽지 않으면 소켓을 닫습니다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	300

다음은 설정값에 대한 설명입니다.

설정값	설명
0	유휴 상태에 따른 시간을 제한하지 않습니다.

tcp/tcp_servers/initial_connection_timeout

사용자로부터 첫 번째 요청을 기다리기 위한 타임아웃을 설정합니다.

TCP 연결을 맺은 후 설정된 시간 동안 사용자가 어떠한 요청도 보내지 않으면 해당 소켓을 닫습니다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	10

다음은 설정값에 대한 설명입니다.

설정값	설명
0	첫 번째 요청을 기다리는 시간이 제한되지 않습니다.

2.10.2. 설정 예시

다음은 SERVER 절을 설정한 예시입니다.

```
"server": {
    "http": {
        "common_config": {
            "doc_root": "docs",
            "idle_timeout": 300,
            "service_order": "uri,ext",
            "access_log": "access_log1",
            "keepalive": true,
            "keepalive_timeout": 60
        },
        "http_servers": [
            {
                "name": "http1",
                "port": 8080,
                "enable_ssl": false
            }
        ]
    }

    "wjp": {
        "port": 9900,
        "wjp_servers": [
            {
                "name": "MyGroup1",
                "svr_chk_time": 60
            }
        ]
    }

    "tcp": {
        "tcp_servers": [
            {
                "name": "tcp1",
                "port": 5000,
                "server_address": [
                    "192.168.1.20:5000"
                ]
            }
        ]
    }
}
```

2.11. SERVICE 절

HTTP 요청을 처리할 내부 서버를 요청의 URI, EXT 기반으로 설정합니다.

2.11.1. 설정 항목

다음은 SERVICE 절의 환경 설정 형식입니다.

```
"service": {
    "uri": [
```

```
        {
            "name": string,
            #"target_http_servers": [string],       # ["*"]
            "match": {
                #"rewrite": string,
                #"type": string,                    # "prefix"
                "target": string,
                #"redirect": string,
                #"redirect_status": integer,        # 302
                #"enable_cache": boolean            # false
            },
            "destination": {
                "type": string,
                "target": string
            },
            #"access": string
        }
    ]
    "ext": [
        {
            "name": string,
            #"target_http_servers": [string],       # ["*"]
            "match": {
                #"type": string,                    # "exact"
                "target": string,
                #"enable_cache": boolean            # false
            },
            "destination": {
                "type": string,
                "target": string
            },
            #"access": string
        }
    ]
}
```

절과 설정 항목의 구성에 대한 기호나 내용에 대한 자세한 내용은 설정 항목 값의 형식 및 설정 방법을 참고합니다.

uri

URI 절은 클라이언트 요구의 URI(Uniform Resource Identifier) 값에 따라 요청을 처리할 목적지(destination)를 구분할 수 있도록 설정합니다.

예를 들어 사용자가 http://www.tmax.co.kr/JSV/test를 호출한 경우 "/JSV/" URI를 jeus로 설정하여 처리할 수 있습니다.

구분	설명
자료형	array(object)

아래와 같은 요청이 있을 때는 <first>로 시작하는 URI로 설정해야 합니다.

```
http://<hostname>:<port>/<first>/<second>/index.html
```

uri/name (필수 항목)

URI 절의 이름을 설정합니다.

구분	설명
자료형	string
범위	31자 이내

uri/target_http_servers

특정 서버가 처리하도록 설정할 경우 해당 서버의 이름을 설정합니다. (예: "target_http_servers": "http1")

만약 특정 vhost가 처리하도록 설정할 경우에는 "서버의 이름.vhost 이름"으로 설정합니다. (예: "target_http_servers": "http1.vhost1")

구분	설명
자료형	array[string]
범위	64개 이내[31자 이내]
기본값	["*"]

다음은 설정값에 대한 설명입니다.

설정값	설명
"*"	모든 서버에 설정됩니다.

uri/match (필수 항목)

요청의 URI가 특정 패턴과 일치하는 경우 해당 요청을 처리할 규칙을 설정합니다.

구분	설명
자료형	object

uri/match/rewrite

uri/match/target을 어떤 값으로 변경할지 설정합니다.

구분	설명
자료형	string

구분	설명
범위	255자 이내

uri/match/type

URI에 설정된 패턴의 유형을 설정합니다. 패턴 유형에 따라 HTTP 요청 경로와 매치하는 방식이 달라집니다.

구분	설명
자료형	string
범위	"prefix"
기본값	"prefix"

다음은 설정값에 대한 설명입니다.

설정값	설명
prefix	uri/match/target에 설정된 패턴이 HTTP 요청 URL의 접두사(prefix)입니다. (예: 패턴 "/uri/"는 "/uri/a/", "/uri/a/b/", "/uri/a/b/c" 등의 요청 경로와 매치)

uri/match/target (필수 항목)

HTTP 요청 경로와 매치할 패턴을 설정합니다. 매치되면 해당 요청은 URI 절의 설정이 적용됩니다.

구분	설명
자료형	string
범위	255자 이내

uri/match/redirect

지정된 URI 요청을 다른 URI로 매핑하도록 설정합니다.

URI 절에 설정된 redirect_status의 값에 따라 redirect에 설정된 값이 HTTP 응답의 Location 헤더 필드에 설정되어 사용자에게 전달됩니다. 만약 redirect_status의 값을 생략한 경우 기본값으로 "302 Found"가 사용됩니다.

구분	설명
자료형	string
범위	255자 이내

uri/match/redirect_status

다른 URI로 리다이렉트 시 사용할 HTTP 상태 코드를 설정합니다.

구분	설명
자료형	integer
범위	301 | 302 | 303 | 305 | 407 | 410
기본값	302

다음은 설정값에 대한 설명입니다.

설정값	별칭	설명
301	permanent	"301 Moved Permanently"로 응답합니다.
302	found	"302 Found"로 응답합니다.
303	seeother	"303 See Other"로 응답합니다.
305	useproxy	"305 Use Proxy"로 응답합니다.
307	temp	"307 Temporary Redirect"로 응답합니다.
410	gone	"410 Gone"로 응답합니다.

uri/match/enable_cache

콘텐츠의 캐시 유무를 설정합니다.

구분	설명
자료형	boolean
기본값	false

uri/destination (필수 항목)

URI 서비스의 목적지를 설정합니다.

구분	설명
자료형	object

uri/destination/type (필수 항목)

목적지의 유형을 설정합니다.

구분	설명
자료형	string
범위	"HTMLS" | "JEUS" | "REVERSE_PROXY"

uri/destination/target (필수 항목)

목적지의 대상을 설정합니다. 이때 DESTINATION 절에 설정된 name과 동일해야 합니다.

구분	설명
자료형	string
범위	127자 이내

uri/match/access

특정 IP에서 들어온 요청에 대해 허용 여부를 설정합니다. 이때 설정값은 access/access_list/name에 설정된 이름과 일치해야 합니다.

구분	설명
자료형	string
범위	255자 이내

ext

클라이언트가 요구한 파일의 확장자명에 따라 처리 담당 프로세스를 설정합니다.

구분	설명
자료형	array(object)

WebtoB는 기본적인 모든 MIME-Type에 대한 처리 담당 프로세스가 설정되어 있으나, 추가적인 설정이 필요할 경우 해당 절에서 수정할 수 있습니다.

ext/name (필수 항목)

EXT 절의 이름을 설정합니다.

구분	설명
자료형	string
범위	31개 이내

ext/target_http_servers

특정 서버가 처리하도록 설정할 경우 해당 서버의 이름을 설정합니다. (예: "target_http_servers": "http1")

만약 특정 vhost가 처리하도록 설정할 경우에는 "서버의 이름.vhost 이름"으로 설정합니다. (예: "target_http_servers": "http1.vhost1")

구분	설명
자료형	array(string)
범위	64개 이내(31자 이내)
기본값	["*"]

다음은 설정값에 대한 설명입니다.

설정값	설명
"*"	모든 서버에 설정됩니다.

ext/match (필수 항목)

파일의 확장자명이 특정 패턴과 일치하는 경우 해당 파일을 처리할 규칙을 설정합니다.

구분	설명
자료형	object

ext/match/type

ext/match/target에 설정된 패턴의 유형을 설정합니다. 패턴 유형에 따라 HTTP 요청 경로와 매치하는 방식이 달라집니다.

구분	설명
자료형	string
범위	"exact"
기본값	"exact"

다음은 설정값에 대한 설명입니다.

설정값	설명
exact	설정된 패턴이 확장자와 일치하면 매치합니다. (예: 패턴 "abc"는 확장자가 "abc"일 경우만 매치합니다. 이외 모든 확장자는 매치하지 않습니다.)

ext/match/target (필수 항목)

HTTP 요청 경로의 확장자와 매치할 패턴을 설정합니다. 매치되면 해당 요청은 EXT 절의 설정이 적용됩니다.

구분	설명
자료형	string

구분	설명
범위	127개 이내

ext/match/enable_cache

콘텐츠의 캐시 유무를 설정합니다.

구분	설명
자료형	boolean
기본값	false

ext/destination (필수 항목)

Extension 서비스의 목적지를 설정합니다.

구분	설명
자료형	string

ext/destination/type (필수 항목)

목적지의 유형을 설정합니다.

구분	설명
자료형	string
범위	"HTMLS" | "JEUS" | "REVERSE_PROXY"

ext/destination/target (필수 항목)

목적지의 대상을 설정합니다. 이때 DESTINATION 절에 설정된 name과 동일해야 합니다.

구분	설명
자료형	string
범위	127자 이내

ext/match/access

특정 IP에서 들어온 요청에 대해 허용 여부를 설정합니다. 이때 설정값은 access/access_list/name에 설정된 이름과 일치해야 합니다.

구분	설명
자료형	string

구분	설명
범위	255자 이내

2.11.2. 설정 예시

다음은 SERVICE 절을 설정한 예시입니다.

```
"service": {
    "uri": [
        {
            "name": "static_uri",
            "target_http_servers": [
                "*"
            ],
            "match": {
                "type": "prefix",
                "target": "/static",
                "rewrite": "/"
            },
            "destination": {
                "type": "HTMLS",
                "target": "htmls1"
            }
        }
    ],
    "ext": [
        {
            "name": "ext1",
            "target_http_servers": [
                "http1.vhost1"
            ],
            "match": {
                "type": "exact",
                "target": "text/html"
            },
            "destination": {
                "type": "HTMLS",
                "target": "html1"
            }
        }
    ]
}
```

2.12. SSL 절

WebtoB에서 사용할 SSL의 기능을 설정합니다. 해당 절에 정의된 형태로 SSL 서비스를 제공합니다.

2.12.1. 설정 항목

다음은 SSL 절의 환경 설정 형식입니다.

```
#"ssl": {
    #"common_config": {                                 # COMMON
        #"verify_depth": integer,                       # 0 (0-INT_MAX)
        #"protocols": [string],                         # ["TLSv1", "TLSv1.1", "TLSv1.2", "TLSv1.3"]
        #"required_ciphers": string,                    # "HIGH:!RSA"
        #"tls13_required_ciphers": string               #
"TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256:TLS_AES_128_GCM_SHA256"
    },
    #"ssl_configs": [
        {
            "name": string,
            "certificate_file": string,
            "certificate_key_file": string,
            #"certificate_chain_file": string,
            #"certificate_key_password": string,        # "builtin"
            #"ca_certificate_file": string,
            #"ca_certificate_path": string,
            #"ssl_server_name": [string],
            #"verify_client": integer,                  # 0 (0-3)
            #"renegotiation_level": string,             # "secure"
            #"enable_stapling": boolean,                # false
            #"common_config": {...}                     # COMMON
        }
    ],
    #"proxy_ssl_configs": [
        {
            "name": string,
            #"proxy_certificate_file": string,
            #"proxy_certificate_key_file": string,
            #"proxy_certificate_chain_file": string,
            #"proxy_certificate_key_password": string,  # "builtin"
            #"proxy_ca_certificate_file": string,
            #"proxy_ca_certificate_path": string,
            #"ssl_server_name": [string],
            #"enable_insecure": boolean,                # false
            #"common_config": {...}                     # COMMON
        }
    ]
}
```

절과 설정 항목의 구성에 대한 기호나 내용에 대한 자세한 내용은 설정 항목 값의 형식 및 설정 방법을 참고합니다.

common_config

SSL 절의 공통적인 설정입니다. 상위 항목에 설정하면 하위 항목에는 설정하지 않아도 상위 항목의 설정이 반영됩니다.

구분	설명
자료형	object

구분	설명
설정 우선순위	설정할 때 적용되는 우선순위는 다음과 같습니다. 1. "ssl_configs", "proxy_ssl_configs" 2. "ssl"

common_config/verify_depth

인증 시 연결된 CA들의 깊이를 얼마나 깊게 추적할지를 설정합니다. 단 하나의 인증 CA만 필요하다면 1로 설정합니다.

구분	설명
자료형	integer
범위	0 ~ INT_MAX
기본값	0

common_config/protocols

서버가 사용할 수 있는 프로토콜을 설정합니다. 이때 특정 TLS 버전에 대한 지원 여부를 설정할 수 있으며, 특정 프로토콜을 사용하지 않을 경우 프로토콜 이름 앞에 하이픈(-)을 붙여 설정합니다.

구분	설명
자료형	array(string)
범위	1개 이상 4개 이내("TLSv1" | "TLSv1.1" | "TLSv1.2" | "TLSv1.3")
기본값	["TLSv1", "TLSv1.1", "TLSv1.2", "TLSv1.3"]

"SSLv2", "SSLv3"는 지원하지 않습니다.

common_config/required_ciphers

서버가 사용할 수 있는 cipher를 설정합니다. 이때 특정 cipher와 SSL/TLS 버전에 대한 지원 여부를 설정할 수 있습니다.

구분	설명
자료형	string
범위	1023자 이내
기본값	"HIGH:!RSA"

WebtoB는 OpenSSL을 사용하기 때문에 cipher 이름은 OpenSSL 설명서를 참고합니다.

common_config/tls13_required_ciphers

사용할 수 있는 TLS 1.3 cipher를 설정합니다. 이때 특정 cipher와 SSL/TLS 버전에 대한 지원 여부를 설정할 수 있습니다.

구분	설명
자료형	string
범위	1023자 이내
기본값	"TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256:TLS_AES_128_GCM_SHA256"

WebtoB는 OpenSSL을 사용하기 때문에 cipher 이름은 OpenSSL 설명서를 참고합니다.

ssl_configs

WebtoB가 SSL 서버 역할로 동작할 때의 설정입니다.

구분	설명
자료형	array(object)
범위	100개 이내

ssl_configs/name (필수 항목)

SSL 서버 설정의 이름입니다. 다른 절에서 SSL 서버 설정을 사용할 때는 이 'name'을 설정해야 합니다.

구분	설명
자료형	string
범위	31자 이내

ssl_configs/certificate_file (필수 항목)

PEM 방식으로 인코딩된 서버 인증서를 설정합니다.

PEM은 DER 규칙으로 인코딩되어 ASCII 형식으로 웹에서 전송됩니다. 만약 인증서가 암호화된 상태일 경우 비밀번호(passphrase)를 입력해야 합니다.

구분	설명
자료형	string
범위	255자 이내

ssl_configs/certificate_key_file (필수 항목)

서버에서 사용되는 PEM 방식으로 인코딩된 인증서의 개인 Key를 설정합니다.

인증서와 개인 Key가 함께 조합되지 않은 경우 이 지시자를 사용하여 Key의 위치를 지정해야 합니다. 일반적으로 WebtoB의 SSL 디렉터리에 위치시킵니다.

구분	설명
자료형	string
범위	255자 이내

ssl_configs/certificate_chain_file

서버 인증서의 인증서 연쇄(Certificate Chain)를 구성할 때 필요한 상위 인증 기관의 인증서 경로를 설정합니다. 단, 클라이언트 인증을 위해서는 ca_certificate_file 또는 ca_certificate_path에 설정해야 합니다.

구분	설명
자료형	string
범위	255자 이내

ssl_configs/certificate_key_password

SSL을 사용하는 경우 암호화된 개인 Key 파일에 대한 암호문을 얻기 위한 방식을 설정합니다.

구분	설명
자료형	string
기본값	"builtin"

다음은 설정값에 대한 설명입니다.

설정값	내용
builtin	WebtoB가 기동될 때 암호문을 입력할 것을 요구합니다.
exec:<program path>	WebtoB가 기동될 때 해당 프로그램을 실행하고, 그 출력 결과를 암호문으로 사용합니다. exec로 실행되는 파일은 컴파일된 실행 파일이나 셸 스크립트가 이용될 수 있습니다.
raw:<password>	WebtoB가 기동될 때 해당 password를 암호문으로 사용합니다.
file:<password file path>	WebtoB가 기동될 때 mkpwd 툴을 이용하여 생성된 password 파일을 이용하여 암호문으로 사용합니다.

다음은 certificate_key_password 설정 예시입니다.

```
"ssl_configs":[{
  "name":"ssl1",
  "certificate_file":"/home/webtob6/ssl/server.crt",
  "certificate_key_file":"/home/webtob6/ssl/server.key",
  "certificate_key_password":"exec:/home/webtob6/ssl/password.sh"
}],
```

ssl_configs/ca_certificate_file

단일 CA로부터 사용자 인증만 받고 싶다면 이 지시자를 사용하여 단일 PEM으로 인코딩된 인증 파일을 설정합니다.

구분	설명
자료형	string
범위	255자 이내

ssl_configs/ca_certificate_path

인증서를 저장할 디렉터리를 설정합니다. 인증서는 사용자 인증을 위해 필요한 내용을 담고 있으며, 일반적으로 PEM 방식으로 인코딩되어야 합니다.

구분	설명
자료형	string
범위	255자 이내

ssl_configs/ssl_server_name

SSL에서 alias로 사용할 수 있는 서버 이름을 설정합니다.

구분	설명
자료형	array(string)
범위	100개 이내

ssl_configs/verify_client

SSL 클라이언트에게 요청할 인증 레벨을 설정합니다.

구분	설명
자료형	integer
범위	0 ~ 3
기본값	0

다음은 설정값에 대한 설명입니다.

설정값	설명
0	아무런 인증 요청을 하지 않습니다.
1	사용자는 사용 가능한 인증 정보를 서버에 제공해야 합니다.
2	사용 가능한 인증 정보를 반드시 서버에 제공해야 합니다.
3	사용자는 사용 가능한 인증 정보를 제공해야 하며, 만일 서버가 인증서를 가지고 있지 않은 상황에서는 인증서 인증 과정이 필요 없습니다.

ssl_configs/renegotiation_level

SSL을 사용하는 경우 재협상(Renetotiation)의 레벨을 설정합니다.

구분	설명
자료형	string
범위	"secure" | "insecure" | "disable"
기본값	"secure"

다음은 설정값에 대한 설명입니다.

설정값	내용
secure	클라이언트와 웹 서버가 모두 안전한 경우 재협상을 진행합니다. (예: RFC5746)
insecure	클라이언트와 웹 서버가 안전하지 않더라도 재협상을 진행합니다. (예: CVE-2009-3555)
disable	어떠한 경우에도 재협상을 진행하지 않습니다.

안전하지 않은 경우 재협상이 진행되면 MITM(Man in the Middle) 공격 또는 DoS(Denial of Service) 공격에 취약할 수 있기 때문에 주의해야 합니다.

ssl_configs/enable_stapling

WebtoB SSL이 OCSP(Online Certificate Status Protocol) stapling 방식으로 동작할지 여부를 설정합니다.

구분	설명
자료형	boolean
기본값	false

ssl_configs/common_config

SSL 절의 공통적인 설정입니다. 상위 항목에 설정하면 하위 항목에는 설정하지 않아도 상위 항목의 설정이

반영됩니다.

구분	설명
자료형	object
설정 우선순위	설정할 때 적용되는 우선순위는 다음과 같습니다. 1. "ssl_configs", "proxy_ssl_configs" 2. "ssl"

proxy_ssl_configs

WebtoB가 SSL 클라이언트 역할로 동작할 때의 설정입니다. WebtoB가 역방향 프록시로 동작하면서 SSL 통신을 할 때 사용됩니다.

구분	설명
자료형	array(object)
범위	100개 이내

proxy_ssl_configs/name(필수 항목)

SSL 클라이언트 설정의 이름입니다. 역방향 프록시 설정에서 SSL를 사용할 때는 이 'name'을 설정해야 합니다.

구분	설명
자료형	string
범위	31자 이내

proxy_ssl_configs/enable_insecure

내부 서버의 인증서가 유효하지 않을 때 SSL 연결을 허용할지 여부를 설정합니다.

구분	설명
자료형	boolean
기본값	false

다음은 설정값에 대한 설명입니다.

설정값	설명
true	SSL 연결을 허용합니다.

proxy_ssl_configs/proxy_certificate_file

PEM 방식으로 인코딩된 클라이언트 인증서를 설정합니다. 내부 서버가 클라이언트 인증을 요구하는 경우 반드시 설정해야 합니다.

PEM은 DER 규칙으로 인코딩되어 ASCII 형식으로 웹에서 전송됩니다. 만약 인증서가 암호화된 상태일 경우 비밀번호(passphrase)를 입력해야 합니다.

구분	설명
자료형	string
범위	255자 이내

proxy_ssl_configs/proxy_certificate_key_file

클라이언트 인증 수행 시 사용되는 PEM 방식으로 인코딩된 인증서의 개인 Key를 설정합니다. 내부 서버가 클라이언트 인증을 원하는 경우 반드시 설정해야 합니다.

인증서와 개인 Key가 함께 조합되지 않은 경우 이 지시자를 사용하여 Key의 위치를 지정해야 합니다. 일반적으로 WebtoB의 SSL 디렉터리에 위치시킵니다.

구분	설명
자료형	string
범위	255자 이내

proxy_ssl_configs/proxy_certificate_chain_file

클라이언트 인증서의 인증서 연쇄(Certificate Chain)를 구성할 때 필요한 상위 인증기관의 인증서 경로를 설정합니다.

구분	설명
자료형	string
범위	255자 이내

proxy_ssl_configs/proxy_certificate_key_password

PROXY_SSL에서 클라이언트 인증(proxy_certificate_file, proxy_certificate_key_file)을 사용하는 경우 암호화된 개인 Key 파일에 대한 암호문을 얻기 위한 방식을 설정합니다.

구분	설명
자료형	string
기본값	"builtin"

다음은 설정값에 대한 설명입니다.

설정값	내용
builtin	WebtoB가 기동될 때 암호문을 입력할 것을 요구합니다.
exec:<program path>	WebtoB가 기동될 때 해당 프로그램을 실행하고, 그 출력 결과를 암호문으로 사용합니다. exec로 실행되는 파일은 컴파일된 실행 파일이나 셸 스크립트가 이용될 수 있습니다.
raw:<password>	WebtoB가 기동될 때 해당 password를 암호문으로 사용합니다.
file:<password file path>	WebtoB가 기동될 때 mkpwd 툴을 이용하여 생성된 password 파일을 이용하여 암호문으로 사용합니다.

proxy_ssl_configs/proxy_ca_certificate_file

단일 CA로부터 사용자 인증만 받고 싶다면 이 지시자를 사용하여 단일 PEM으로 인코딩된 인증 파일을 설정합니다.

구분	설명
자료형	string
범위	255자 이내

proxy_ssl_configs/proxy_ca_certificate_path

인증서를 저장할 디렉터리를 설정합니다. 인증서는 연결할 서버의 인증서를 인증하기 위해 필요한 내용을 담고 있으며, 일반적으로 PEM 방식으로 인코딩되어야 합니다.

구분	설명
자료형	string
범위	255자 이내

proxy_ssl_configs/ssl_server_name

SSL에서 alias로 사용할 수 있는 서버 이름을 설정합니다.

구분	설명
자료형	array(string)
범위	100개 이내(255자 이내)

proxy_ssl_configs/common_config

SSL 절의 공통적인 설정입니다. 상위 항목에 설정하면 하위 항목에는 설정하지 않아도 상위 항목의 설정이 반영됩니다.

구분	설명
자료형	object

구분	설명
설정 우선순위	설정할 때 적용되는 우선순위는 다음과 같습니다. 1. "ssl_configs", "proxy_ssl_configs" 2. "ssl"

2.12.2. 설정 예시

다음은 SSL 절을 설정한 예시입니다.

```
{
    "ssl": {
        "ssl_configs": [
            {
                "name": "ssl1",
                "certificate_file": "server.crt",
                "certificate_key_file": "server_key.crt",
                "certificate_chain_file": "server_chain.crt",
                "certificate_key_password": "builtin",
                "ca_certificate_file": "server_ca.crt",
                "ssl_server_name": [
                    "example.com"
                ],
                "verify_client": 0,
                "renegotiation_level": "secure"
            }
        ],
        "proxy_ssl_configs": [
            {
                "name": "proxy_ssl1",
                "proxy_certificate_file": "server.crt",
                "proxy_certificate_key_file": "server_key.crt",
                "proxy_certificate_chain_file": "server_chain.crt",
                "proxy_certificate_key_password": "builtin",
                "proxy_ca_certificate_file": "internal_server_ca.crt"
            }
        ]
    }
}
```

3. URL Rewrite 기능

3.1. 기능 활성화 설정 방법

URLRewrite 기능을 사용하기 위해서는 /server/http의 common_config에서 "url_rewrite_config" 항목에 Condition과 Rule에 관련된 설정을 해야 합니다.

다음은 URLRewrite 기능을 사용하기 위한 /server/http 설정에 대한 예시입니다.

```
"server": {
    "http": {
        "common_config": {
            "url_rewrite_config": "config/rewrite.conf",
            ...
```

url_rewrite_config에는 Apache의 mod_rewrite의 기능 중 RewriteCond와 RewriteRule 설정을 사용할 수 있습니다. 단, 모든 기능이 지원되는 것은 아니며, 일부 기능은 사용이 제한됩니다.

> 안내서에서 설명하는 URLRewriteConfig 설정은 Apache 2.2 버전의 "mod_rewrite"를 참고하여 작성되었습니다. 설명된 내용 중 일부는 WebtoB에서 동작하지 않을 수 있으므로 유의해야 합니다.

3.2. Rewriting 조건 정의

RewriteCond는 재작성(rewriting) 조건을 정의하며, 다음과 같은 형식으로 설정합니다. TestString과 CondPattern을 매칭하여 조건이 만족되면 RewriteRule 설정에 따라 해당 패턴을 교체합니다.

```
RewriteCond <TestString> <CondPattern> [flags]
```

3.2.1. TestString 설정

TestString에는 다음과 같은 예약어와 일반 문자열을 사용할 수 있습니다.

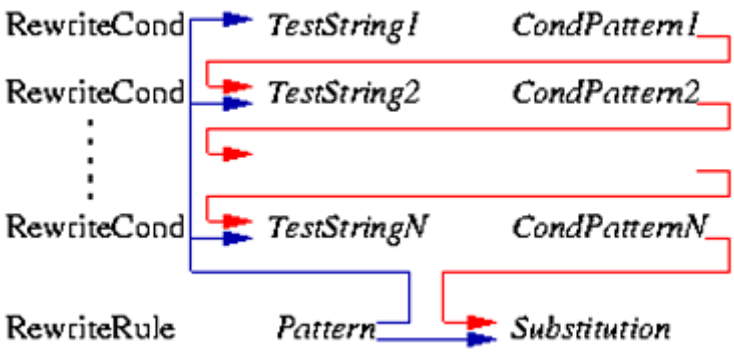
- **$N (0 <= N <= 9)**

  RewriteRule의 Pattern 중 괄호로 묶인 패턴을 참조합니다.

- **%N (1 <= N <= 9)**

  RewriteCond의 CondPattern 중 괄호로 묶인 패턴을 참조합니다.

  $N과 %N은 다음과 같은 Regex Back-Reference 구조를 가집니다.



- **%{SERVER_VARIABLE_NAME}**

  CGI에서 사용하는 서버 환경 변수 중 일부와 몇 가지 추가적인 변수를 참조합니다.

유형	변수
서버 환경 변수	%{ENV:variable}: 환경 변수를 참조, %{HTTP:header}: HTTP 요청 헤더를 참조
HTTP headers	HTTP_USER_AGENT, HTTP_REFERER, HTTP_COOKIE, HTTP_FORWARDED, HTTP_HOST, HTTP_PROXY_CONNECTION, HTTP_ACCEPT
connection & request	REMOTE_ADDR, REMOTE_PORT, REMOTE_METHOD, QUERY_STRING, AUTH_TYPE
server internals	DOCUMENT_ROOT, SERVER_NAME, SERVER_ADDR, SERVER_PORT, SERVER_PROTOCOL, SERVER_SOFTWARE
date and time	TIME_YEAR, TIME_MON, TIME_DAY, TIME_HOUR, TIME_SEC, TIME_WDAY, TIME
specials	THE_REQUEST, REQUEST_URI, HTTPS

Apache의 mod_rewrite에서 제공하는 변수 중 아래의 변수는 사용할 수 없습니다.

```
%{SSL:variable}, %{LA-U:variable}, %{LA-F:variable}, REMOTE_HOST,
REMOTE_IDENT, SCRIPT_FILENAME, SCRIPT_USER, SCRIPT_GROUP, SERVER_ADMIN,
API_VERSION, REQUEST_FILENAME, IS_SUBREQ
```

3.2.2. CondPattern 설정

CondPattern은 Perl compatible reqular expression을 사용할 수 있으며, 다음과 같은 추가적인 패턴을 사용할 수 있습니다.

지시자	설명
!Pattern	Pattern에 매칭되지 않는 경우를 지정합니다.
<CondPattern	TestString이 CondPattern보다 사전 순서상 앞에 있을 경우를 매칭합니다.
>CondPattern	TestString이 CondPattern보다 사전 순서상 뒤에 있을 경우를 매칭합니다.

지시자	설명
=CondPattern	TestString이 CondPattern과 일치하는 경우를 매칭합니다.
-d	TestString을 path로 간주하여 디렉터리일 경우를 매칭합니다.
-f	TestString을 path로 간주하여 파일일 경우를 매칭합니다.
-s	TestString을 path로 간주하여 파일 사이즈가 0보다 큰 경우를 매칭합니다.
-l	TestString을 path로 간주하여 symbolic link일 경우를 매칭합니다.
-x	TestString을 path로 간주하여 실행 파일일 경우를 매칭합니다.
-F	TestString을 path로 간주하여 서버 설정상 접근이 가능한 경우를 매칭합니다.
-U	TestString을 URL로 간주하여 서버 설정상 접근이 가능한 경우를 매칭합니다.

모든 패턴에는 '!'을 붙여서 반대 의미로 매칭할 수 있습니다.

다음은 정규 표현식에서 사용하는 지시자의 설명입니다.

지시자	설명
.	단일 문자와 일치합니다.
+	이전 패턴을 1회 이상 반복합니다.
*	이전 패턴을 0회 이상 반복합니다.
?	이전 패턴을 선택적으로 매칭합니다.
^	이전 패턴을 선택적으로 매칭합니다.
$	문자열의 끝 부분과 일치합니다.
( )	여러 문자를 하나의 그룹으로 묶고, 백레퍼런스에서 사용할 수 있도록 매칭 결과를 캡처합니다.
[ ]	문자 클래스 - 대괄호 안의 문자 중 하나와 일치합니다.
[^]	부정 문자 클래스 - 대괄호 안에 나열된 문자가 아닌 모든 문자와 일치합니다.

3.2.3. flags 설정

CondPattern에 [flags]를 사용하여 패턴 매칭 방식을 변경할 수 있습니다.

지시자	설명
nocase|NC	패턴을 매칭할 때 대소문자를 구분하지 않습니다.
ornext|OR	RewriteCond 다음에 또 다른 RewriteCond가 있을 경우 다음 RewriteCond와 logical OR로 조합되도록 설정합니다. 명시적으로 설정하지 않은 경우 항상 AND 조건으로 다음 조건과 조합됩니다.

지시자	설명
novary|NV	HTTP Header를 TestString으로 사용할 경우 Vary Header가 응답에 추가되지 않도록 합니다.

다음은 [OR] 플래그를 사용한 예시입니다.

```
RewriteCond %{HTTP_HOST}  ^host1.*  [OR]
RewriteCond %{HTTP_HOST}  ^host2.*  [OR]
RewriteCond %{HTTP_HOST}  ^host3.*
RewriteRule ...
```

만약 [OR] 플래그를 사용하지 않는다면 3개의 RewriteCond와 RewriteRule을 각각 별도로 생성해야 합니다.

3.3. Rewriting 동작 정의

RewriteRule은 재작성(rewriting) 동작을 정의하며, 다음과 같은 형식으로 설정합니다. 사용자 요청이 RewriteCond에 매칭되는 경우 RewriteRule 설정에 의해 사용자 요청 중 Pattern을 Substitution으로 교체합니다.

```
RewriteRule <Pattern> <Substitution> [flags]
```

3.3.1. Pattern 설정

Pattern은 Perl compatible reqular expression을 사용할 수 있으며, '!'을 사용하면 패턴에 매칭되지 않는 경우를 지정할 수 있습니다.

'!'을 사용할 경우 괄호로 묶은 그룹 패턴($N)을 사용할 수 없습니다. 이는 패턴이 매칭되지 않아 그룹 패턴($N)에 값이 없기 때문입니다.

RewriteRule만 단독으로 설정할 경우 패턴 매칭은 URL-path를 사용하며, RewriteCond와 함께 사용될 경우 마지막에 매칭된 패턴을 사용합니다.

3.3.2. Substitution 설정

URL을 교체할 값을 설정할 수 있으며, 다음과 같은 값들을 사용할 수 있습니다.

설정값	설명
file-system path	/로 시작하는 file-system의 절대 경로를 설정할 경우 해당 파일을 사용자 응답에 사용합니다. 단, 설정한 경로가 file-system에 존재해야 합니다.
URL-path	일반적인 URL-path를 설정할 경우 해당 리소스를 사용합니다.

설정값	설명
Absolute URL	Absolute URL(예: http://<호스트 이름>/file.html)을 사용할 경우 호스트 이름이 서버와 일치하면 스킴(scheme)과 호스트 이름을 제외한 나머지를 URL-path처럼 사용하고, 일치하지 않을 경우 외부 서버로 리다이렉트합니다.
-	교체하지 않음을 의미합니다.
$N (N=0..9)	RewriteRule의 패턴 중 N번째 그룹 패턴을 지칭합니다.
%N (N=1..9)	마지막에 매칭된 RewriteCond의 패턴 중 N번째 그룹 패턴을 지칭합니다.
%{VARNAME}	서버에서 제공하는 VARNAME을 참조합니다. RewriteCond의 TestString에 적용되는 항목을 사용할 수 있습니다.
?	기본적으로 Query string은 변경되지 않지만, 이를 교체하고 싶을 경우 '?'를 Substitution안에 추가합니다. Query string을 삭제하려면 Substitution의 마지막을 '?'로 끝냅니다.

Apache의 mod_rewrite에서 제공하는 RewriteMap을 지원하지 않기 때문에 '${mapname:key|default}'는 사용할 수 없습니다.

3.3.3. flags 설정

RewriteRule의 세부 동작을 [flags]의 설정을 통해 조절할 수 있습니다.

flags는 콤마(,)를 사용하여 여러 개를 설정할 수 있으며, 다음과 같은 값을 사용할 수 있습니다.

- **B**

  기본적으로 Substitution에 사용되는 backreference($N 또는 %N)는 URL의 %-encoding을 해제하여 사용하지만 [B] 옵션을 사용하면 URL의 %-encoding을 그대로 유지할 수 있습니다.

  예를 들어 아래와 같이 설정된 경우 기본적으로는 "/C%2b%2b"가 "index.php?show=/C++"로 매핑되지만, [B] 옵션을 사용하면 "index.php?show=/C%2b%2b"로 매핑됩니다.

  ```
  RewriteRule ^(.*)$ index.php?show=$1
  ```

- **chain|C**

  현재 rule이 매칭되지 않을 경우 다음 rule을 체크하지 않습니다. 만약 그 다음 rule에도 [C] 옵션이 설정되어 있으면 마찬가지로 건너뛰게 됩니다.

- **cookie|CO=NAME:VAL:domain[:lifetime[:path[:secure[:httponly]]]]**

  응답에 Set-Cookie 헤더를 추가하여 사용자 브라우저에 쿠키를 추가합니다.

- **discardpathinfo|DPI**

  디렉터리별 컨텍스트를 구성한 환경에서 RewriteRule은 URI와 PATH_INFO를 구분하여 매번 rule을 적용할

때 URI와 PATH_INFO를 결합합니다. 이로 인해 URI가 여러 번 매칭될 경우 PATH_INFO가 여러 번 붙게 됩니다.

[DPI] 옵션은 PATH_INFO를 구분하지 않고 RewriteRule을 적용합니다.

- **env|E=VAR:VAL**

  환경 변수에 VAR=VAL을 추가합니다. VAL에는 정규식 백레퍼런스($N 또는 %N)를 사용할 수 있습니다. 이 환경 변수는 SSI나 CGI에서 사용할 수 있으며, RewriteCond의 패턴 중 %{ENV:VAR}으로 사용할 수 있습니다.

- **forbidden|F**

  "403 Forbidden" 응답을 보냅니다.

- **gone|G**

  "410 Gone" 응답을 보냅니다.

- **handle|H=Content-handler**

  Content-handler를 설정합니다.

- **last|L**

  재작성 과정을 이곳에서 종료합니다. C의 break 명령과 유사한 기능을 하는 옵션입니다.

- **next|N**

  지금까지 변경된 URL을 가지고 처음부터 다시 재작성 과정을 수행합니다. C의 continue 명령과 유사한 기능을 하는 옵션입니다.

  해당 옵션 사용 시 무한루프가 발생할 수 있으니 주의해야 합니다.

- **nocase|NC**

  패턴에 대소문자를 구분하지 않도록 합니다. [A-Z]와 [a-z]를 똑같이 취급합니다.

- **noescape|NE**

  재작성 과정에서 URL을 %-encoding하지 않도록 합니다.

- **nosubreq|NS**

  내부 요청일 경우 재작성을 중지합니다.

- **passthrough|PT**

  재작성을 수행한 결과를 다른 핸들러가 사용할 수 있도록 합니다.

- **qsappend|QSA**

  쿼리 문자열을 재작성할 때 기존 쿼리 문자열을 덮어쓰지 않고, 그대로 유지하면서 새로운 쿼리 문자열을 추가합니다.

- **redirect|R[=code]**

  Substitution이 Absolute URL인 경우 호스트 이름이 서버와 일치하더라도 강제로 리다이렉트합니다.

  코드가 지정되지 않으면 기본적으로 302 Moved Temporarily가 사용됩니다. 코드에는 상태 코드를 직접 입력하거나, temp(기본값), permanent, seeother를 설정할 수 있습니다.

- **skip|S=num**

  다음 num개의 rule를 건너뛰도록 합니다.

- **type|T=MIME-type**

  응답의 Content-Type을 설정합니다.

3.4. 상황별 URL Rewrite 설정 예시

다음은 웹 서버에서 발생할 수 있는 다양한 상황에 대한 URLRewriteConfig 설정 예시입니다.

- **상황 1: 기본 URL 리다이렉션**

  다음은 "www.test.com/"과 같은 URL 패턴을 매칭하여, "www.test.com/rewrite.html"로 변환하는 설정 예시입니다.

```
# url rewrite config - ex1
RewriteCond %{HTTP_HOST} ^www\.test\.com$       # if {Host} == "www.test.com"
RewriteRule ^/$ /rewrite.html [L]               # then "/" > "/rewrite.html"
```

- **상황 2: 파일이 존재하지 않을 경우 에러 페이지로 리다이렉션**

  다음은 "www.test.com/temp/xxx.html"과 같은 요청을 할 때 temp 디렉터리에 xxx.html 파일이 없는 경우 "www.test.com/temp/temp_error.html"로 변환하는 설정 예시입니다.

```
# url rewrite config - ex2
RewriteCond %{REQUEST_FILENAME} !-f
# if {요청 파일 이름} != 파일을 가리키거나 포함
RewriteRule ^/([^/]+) /$1/$1_error.html [L]
# then "/temp/xxx.html" > "/temp/temp_error.html"
```

- **상황 3: HTTP 요청을 HTTPS로 리다이렉션**

  다음은 "http://www.test.com:80"과 같은 요청을 "https://www.test.com:443"으로 변환하는 설정 예시입니다.

```
# url rewrite config - ex3
RewriteCond %{HTTP_HOST} ^www\.test\.com$
# if {Host} == "www.test.com"
RewriteCond %{SERVER_PORT} 80
```

```
# AND {Port} == "80"
RewriteRule .* https://www.test.com:443$0 [R]
# then > "https://www.test.com:443$0"   ($0: request uri)
```

- **상황 4: 도메인 변경 (URI 무시)**

  다음은 "www.test.com/xxx.html"과 같은 URL 패턴을 매칭하여 요청 URI는 무시하고 무조건 "www.test_new.com/"으로 변환하는 설정 예시입니다.

```
# url rewrite config - ex4
RewriteCond %{HTTP_HOST} ^www\.test\.com$       # if {Host} == "www.test.com"
RewriteRule .* http://www.test_new.com [R]       # then > "http://www.test_new.com"
```

- **상황 5: 도메인 변경 (URI 유지)**

  다음은 "www.test.com/test.html"과 같은 URL 패턴을 매칭하여 "www.test_new.com/test.html"로 변환하는 설정 예시입니다.

```
# url rewrite config - ex5
RewriteCond %{HTTP_HOST} ^www\.test\.com$
# if {Host} == "www.test.com"
RewriteRule .* http://www.test_new.com$0 [R]
# then > "http://www.test_new.com$0"   ($0: request uri)
```

- **상황 6: POST 요청에서 Referer 헤더 없는 경우 차단**

  다음은 요청 메소드가 POST이고 Referer 헤더가 없을 경우 "403 Forbidden" 에러로 처리하는 설정 예시입니다.

```
# url rewrite config - ex6
RewriteCond %{REQUEST_METHOD} POST              # if {Method} == "POST"
RewriteCond %{HTTP_REFERER} =""                 # AND {Referer} == ""
RewriteRule . - [F]                             # then > 403 Forbidden Return
```

- **상황 7: 서브도메인별 URL 리다이렉션**

  다음은 "aaa.test.com/xxx.html"로 요청할 경우 "www.test.com/aaa/xxx.html"로, "bbb.test.com/xxx.html"로 요청할 경우 "www.test.com/bbb/xxx.html"로 처리하는 설정 예시입니다.

```
# url rewrite config - ex8
RewriteCond %{HTTP_HOST} ^(aaa|bbb)\.test\.com
# if {Host} == ("aaa.test.com" OR "bbb.test.com")
RewriteRule .* /%1$0 [L]
# then "/xxx.html" > "/(aaa|bbb)/xxx.html" (%1: RewriteCond의 첫 번째 괄호, $0: request uri)
```

- **상황 8: 쿼리 문자열에 따른 포트 변경**

다음은 "www.test.com/test.do?query=value1"로 요청할 경우 "www.test.com:8080/test.do?query=value1"로, "www.test.com/test.do?query=value2&.."으로 요청할 경우 "www.test.com:8080/test.do?query=value2&.."로 처리하는 설정 예시입니다.

```
# url rewrite config - ex9
RewriteCond %{QUERY_STRING} ^query=value1$ [OR]
# if {QueryString} == "query=value1"
RewriteCond %{QUERY_STRING} ^query=value2&
# OR {QueryString} == "query=value2&.."
RewriteRule .* http://www.test.com:8080$0 [R]
# then > "http://www.test.com:8080$0"   ($0: request uri)
```

- **상황 9: 특정 서브도메인 요청을 www 도메인으로 리다이렉션**

다음은 "aaa.test.com/test.html"과 같이 www로 시작하지 않는 요청을 "www.test.com/aaa/test.html"로 처리하는 설정 예시입니다.

```
# url rewrite config - ex10
RewriteCond %{HTTP_HOST} !^www\.test\.com$
# if {Host} != www.test.com
RewriteCond %{HTTP_HOST} ^([a-zA-Z0-9]+)\.test.com$
# AND {Host} == "(영문/숫자 조합).test.com"
RewriteRule  .* /%1/$0 [L]
# then > "(영문/숫자 조합)/$0"   (%1: RewriteCond의 첫 번째 괄호, $0: request uri)
```

- **상황 10: Referer 헤더 없는 CSS/JS 요청 차단**

다음은 요청 헤더에 Referer 헤더가 없이 CSS나 JS 파일을 요청하는 경우 403 Forbidden으로 처리하는 설정 예시입니다.

```
# url rewrite config - ex11
RewriteCond %{HTTP_REFERER} !.        # if {HTTP_REFERER} == ""
RewriteRule \.(css|js)$ - [F]         # then "*.css|*.js" > 403 Forbidden Return
```

- **상황 11: 특정 패턴을 URL 쿼리 파라미터로 변환**

다음은 "/user@somehost.com/"과 같은 URL 패턴을 매칭하여 "/req_test.jsp?blogId=user@somehost.com"으로 변환하는 설정 예시입니다.

```
# url rewrite config - ex12 (REQEUST_URI)
RewriteCond %{REQUEST_URI} /([a-zA-Z0-9_-]+(@([a-zA-Z0-9_-]+).(com|net|co.kr))?)/?$
RewriteRule . /req_test.jsp?blogId=%1 [PT,L]
```

- **상황 12: 특정 호스트와 URI 패턴에 따른 리다이렉션**

다음은 HTTP 요청 헤더의 호스트가 "tmaxsoft.com"으로 끝나고, URL이 "/redirect/"로 시작하는 경우를 매칭하여 "http://www.tmaxsoft.com/redirect.html"로 리다이렉션하는 설정 예시입니다.

```
# url rewrite config - ex13 (HTTP_HOST)
RewriteCond %{HTTP_HOST} tmaxsoft.com$
RewriteCond %{REQUEST_URI} /redirect/.*$
RewriteRule . http://www.tmaxsoft.com/redirect.html [R,L]
```

4. WebtoB Admin API

WebtoB는 시스템 동작 중 각종 상태 정보 확인, 제어 등의 관리 기능을 제공합니다. 예를 들어 제공되는 서비스들 중 특정 서비스에 대하여 현재 처리 상태 정보(서비스 처리 건수, 평균 처리 시간, 요청 대기 건수 등)를 확인할 수 있습니다.

WebtoB는 이와 같은 기능들을 Rest API를 이용한 Admin API를 통해 제공합니다. /server/admin 설정에서 포트를 지정하여 해당 포트를 통해 API 통신을 수행합니다.

config에서 admin 설정은 다음과 같이 수행합니다. 아래와 같이 설정하면 이후 9090 포트에 대해 Admin API 통신이 가능합니다.

```
"server":{
    "admin": {
        "port": 9090
    }
}
```

다음은 Admin API가 제공하는 API에 대한 설명입니다.

API 이름	설명
config	환경 설정 내용을 조회합니다.
client-info	접속 웹 브라우저를 확인합니다.
svg-info	서버 정보를 확인합니다.
stat-info	프로세스, 스레드 및 서비스 상태에 대한 통계를 조회합니다.
cache-list	HTTP 응답 캐시에 저장된 응답들의 정보를 출력합니다.

WebtoB Admin API의 주요 특징을 다음과 같습니다.

- 모든 API는 POST 메소드로 통신합니다.
- HTH별로 동작하는 API의 경우 HTH가 일정 시간동안 응답이 없다면 타임아웃 이후 "busy_hth"에 표시됩니다. 예를 들어 /client-info API에 대해 HTH-1, HTH-2가 타임아웃동안 응답하지 않는 경우 해당 정보가 다음과 같이 "busy_hth"에 표현됩니다.

  ```
  {
      "result": {
          "HTH-0": [
              {
                  ...
              }
          ]
      },
      "busy_hth": [
          "HTH-1",
          "HTH-2"
  ```

```
    ]
}
```

- API가 올바르게 동작한 경우 HTTP 응답 코드 200을 반환하고, 그렇지 않은 경우에는 상황에 따라 적절한 응답 코드를 반환합니다.

  공통적으로 발생하는 에러 응답 코드는 다음과 같습니다.

응답 코드	설명
400	HTTP 본문에 대한 JSON 스키마 검증에 실패
401	사용자 인증에 실패
403	대상 API에 대한 동작 권한이 없음
404	대상 API가 없음
405	올바르지 않은 메소드로 요청
500	API 동작 중 에러가 발생 (API별로 발생하는 상황과 메시지가 다름)

- 응답 코드가 200이 아닌 경우 공통적으로 다음과 같은 방식으로 구성됩니다.

```
{
    "error_code": string,
    "error_message": string
}
```

  예를 들어 응답 코드가 404인 경우는 다음과 같이 발생할 수 있습니다.

```
{
    "error_code": "APIERR_COMMON_0003",
    "error_message": "No such API"
}
```

4.1. 환경 정보

4.1.1. /config

현재 동작 중인 시스템의 환경 정보를 조회합니다. 이 과정에서는 설정 파일에서 지정한 값뿐만 아니라 기본적으로 적용되는 값까지 확인할 수 있습니다.

- 사용법

```
{
    "path": string
}
```

옵션	설명
path	config의 JSON 경로를 설정합니다. 경로를 생략하거나 /로 설정하면 현재 설정된 모든 config를 가져옵니다. 경로는 항상 /로 시작해야 합니다.

- 예시

다음은 NODE 절 환경 설정을 "/config" Admin API로 확인한 예시입니다. NODE 절의 설정 항목에 대한 자세한 설명은 NODE 절 설정 항목을 참고합니다.

```
[Request]
{
    "path": "/node"
}

[Response]
{
    "result": {
        "cacheEntry": 128,
        "cacheKey": "HOST_URI",
        "cacheMaxFileSize": 8192,
        "connectionPoolSize": 8192,
        "gracefulShutdownTimeout": 30,
        "hthCount": 1,
        "hthSchedule": "RR",
        "maxCacheMemorySize": 100,
        "name": "webtob_node",
        "webtobDir": "$WEBTOBDIR",
        "workerThreads": 1
    }
}
```

4.2. 동작 상태 정보

4.2.1. /client-info

현재 접속된 클라이언트(주로 웹 브라우저)의 환경 정보를 조회합니다. 현재 상태(status), 접속 IP 주소, 처리 건수(count)와 같은 정보를 확인할 수 있습니다.

- 사용법

```
{
    "virtual_host": string,
    "hth_number": integer
}
```

옵션	설명
(empty)	서버에 연결된 모든 클라이언트의 정보를 조회합니다.

옵션	설명
virtual_host	지정한 가상 호스트에 연결된 클라이언트의 정보를 조회합니다.
hth_number	지정한 HTH에 연결된 클라이언트의 정보를 조회합니다. 지정하지 않으면 모든 HTH에 대해 조회합니다.

- 예시

다음은 HTH 3번에서 vhost1 가상 호스트에 연결된 클라이언트 정보를 "/client-info" Admin API로 확인한 예시입니다.

```
[Request]
{
    "virtual_host": "vhost1",
    "hth_number": 3
}

[Response]
{
    "result": {
        "HTH-3": [
            {
                "count": 0,
                "idle": 24,
                "local_ipaddr:port": "192.168.15.167:8080",
                "no": 1,
                "remote_ipaddr:port": "192.168.15.168:48006",
                "ssl": false,
                "status": "READY"
            },
            {
                "count": 2,
                "idle": 24,
                "local_ipaddr:port": "192.168.15.167:8080",
                "no": 0,
                "remote_ipaddr:port": "192.168.15.168:48005",
                "ssl": false,
                "status": "READY"
            }
        ]
    }
}
```

다음은 출력 항목에 대한 설명입니다.

출력 항목	설명
count	해당 클라이언트가 전송한 요청 수
idle	해당 클라이언트가 어떠한 데이터도 주고받지 않고 있는 상태로 지속된 시간
local_ipaddr:port, remote_ipaddr:port	서버와 클라이언트 IP:PORT

출력 항목	설명
no	WebtoB 내부적으로 관리하는 커넥션 번호
ssl	해당 클라이언트가 SSL로 연결되어 있는지 여부
status	서버 내부의 클라이언트 상태 ◦ READY: 클라이언트로부터 요청을 받는 중 ◦ RUNNING: 클라이언트의 요청이 서버에서 처리 중

4.2.2. /svg-info

현재 동작 중인 각 서버 그룹의 정보를 조회합니다.

- 사용법

```
{
    "jeus" : array(string),
    "rproxy" : array(string)
}
```

옵션	설명
jeus	JEUS 서버 그룹 이름을 설정합니다. "*"을 포함하면 모든 서버에 대한 정보를 확인할 수 있습니다.
rproxy	역방향 프록시 그룹 이름을 설정합니다. "*"을 포함하면 모든 서버에 대한 정보를 확인할 수 있습니다.

- 예시

  다음은 모든 JEUS 서버 그룹과 rproxy1이라는 이름의 역방향 프록시 그룹 정보를 "/svg-info" Admin API로 확인한 예시입니다.

```
[Request]
{
    "jeus" : [*],
    "rproxy" : ["rproxy1"]
}

[Response]
"result": [
    {
        "HTH-0": {
            "jeus": [
                {
                    "aqcnt": 0,
                    "count": 0,
                    "cqcnt": 0,
                    "qpcnt": 0,
                    "reqs": 0,
```

```
                    "rscnt": 0,
                    "status": "NRDY",
                    "svgname": "MyGroup2"
                },
                {
                    "aqcnt": 0,
                    "count": 0,
                    "cqcnt": 0,
                    "qpcnt": 0,
                    "reqs": 0,
                    "rscnt": 0,
                    "status": "NRDY",
                    "svgname": "MyGroup1"
                }
            ],
            "rproxy": [
                {
                    "aqcnt": 0,
                    "count": 0,
                    "cqcnt": 0,
                    "qpcnt": 0,
                    "reqs": 0,
                    "rscnt": 0,
                    "status": "RDY",
                    "svgname": "rproxy1"
                }
            ]
        }
    }
]
```


다음은 출력 항목에 대한 설명입니다.

출력 항목	설명
aqcnt	현재까지 큐에 대기했던 요청 수(cqcnt의 cumulative 값)
count	요청 처리 수
cqcnt	현재 큐에서 대기 중인 요청 수
qpcnt	큐에 대기 중이던 요청이 timeout 또는 qp 명령 등으로 인해 큐에서 제거된 요청 수
reqs	해당 서버에 보내진 요청 수
rscnt	해당 서버의 비정상 종료로 인한 재시작 횟수
status	서버 내부의 클라이언트 상태 • RDY: 서버가 요청을 처리할 수 있으며, WebtoB와 연결된 서버 프로세스들이 존재함 • NRDY: 요청을 처리할 수 없으며, WebtoB와 연결된 서버 프로세스가 존재하지 않음 • BLK: 서버가 관리자 명령에 따라 일시 정지된 상태이며, 이로 인해 요청을 처리할 수 없음

출력 항목	설명
svgname	환경 설정의 서버 그룹 이름

4.2.3. /stat-info

실질적인 시스템 동작 상태를 나타내며, 동작 중인 서버 프로세스와 서비스에 대한 정보를 조회합니다.

서버별로 처리한 서비스 개수, 커넥션 개수 등과 같은 동적인 정보를 확인할 수 있습니다.

- 사용법

```
{
    "hth_number" : integer,
    "target":{
        "jeus": array(string),
        "rproxy" : array(string),
        "htmls": boolean
    }
}
```

옵션	설명
hth_number	통계를 확인할 HTH를 지정합니다.
target/jeus	JSV 서버들의 통계 정보를 출력합니다. "*"을 포함하면 모든 서버에 대한 정보를 확인할 수 있습니다.
target/rproxy	역방향 프록시의 각 서버별 상태를 출력합니다. "*"을 포함하면 모든 서버에 대한 정보를 확인할 수 있습니다.
target/htmls	HTMLS의 통계 정보 출력 여부입니다.

target에서 최소한 1개 이상의 항목은 지정해야 합니다.

- 예시

다음은 MyGroup1이라는 JEUS 서버, rproxy1이라는 이름의 역방향 프록시 그룹 내 서버, HTMLS 서버 정보를 "/stat-info" Admin API로 확인한 예시입니다.

```
[Request]
{
    "hth_number" : 1,
    "target":{
        "jeus": ["MyGroup1"],
        "rproxy" : ["rproxy1"],
        "htmls": true
    }
}
```

```
[Response]
{
    "result": {
        "HTH-0": {
            "jeus": [
                {
                    "average_processed_time": 0,
                    "connections": 50,
                    "request_count": 0,
                    "response_count": 0,
                    "server": "amV1c19kb21haW4vc2VydmVyMQ==",
                    "server_group": "MyGroup1",
                    "sticky_routed_count": 0
                }
            ],
            "rproxy": [
                {
                    "average_processed_time": 0,
                    "connections": 0,
                    "request_count": 0,
                    "response_count": 0,
                    "server": "192.168.15.167:8090",
                    "server_group": "rproxy1",
                    "sticky_routed_count": 0
                },
                {
                    "average_processed_time": 0,
                    "connections": 0,
                    "request_count": 0,
                    "response_count": 0,
                    "server": "192.168.15.167:8088",
                    "server_group": "rproxy1",
                    "sticky_routed_count": 0
                }
            ],
            "htmls": {
                "average_processed_time": 0.000309519,
                "request_count": 5,
                "response_count": 5,
                "server": "HTMLS"
            }
        }
    }
}
```

다음은 출력 항목에 대한 설명입니다.

출력 항목	설명
average_processed_time	요청당 평균 처리 시간(초)
connections	해당 서버에 연결된 커넥션 개수
request_count	해당 프로세스로 보내진 요청 수
response_count	해당 프로세스가 처리한 요청 수

출력 항목	설명
server	서버 그룹에 속한 서버 이름 ◦ JEUS: jenginedid ◦ 역방향 프록시: 대상 서버 주소 ◦ HTMLS: HTMLS (고정)
server_group	서버 그룹 이름
sticky_routed_count	클라이언트로부터 받은 스티키(Sticky) ID를 기준으로 해당 서버에 라우팅한 횟수

4.3. 캐시 정보 관리

4.3.1. /cache-list

현재 WebtoB의 HTTP 응답 캐시에 저장된 응답 정보를 출력합니다.

- 사용법

```
{
    "hth_number": integer
}
```

옵션	설명
"hth_number"	캐시 정보를 조회할 HTH를 설정합니다. 지정하지 않으면 모든 HTH의 정보를 조회합니다.

- 예시

다음은 HTH 0번에 대한 캐시 정보를 "/cache-list" Admin API로 확인한 예시입니다. 요청 경로 외의 부분은 서버 내부 디버그 용도로만 사용됩니다.

```
[Request]
{
    "hth_number": 0
}


[Response]
"result": {
    "HTH-0": {
        "cache_map": [
            {
                "expire_time": "2023-10-25 16:35:18",
                "key": "192.168.15.167:8080/index.html",
                "size": 4312
            },
```

```
            {
                "expire_time": "2023-10-25 16:34:52",
                "key": "192.168.15.167:8080/favicon.ico",
                "size": 4507
            },
            {
                "expire_time": "2023-10-25 16:34:52",
                "key": "192.168.15.167:8080/4.html",
                "size": 219
            }
        ],
        "cache_map_size": 3,
        "memory_usage": 9038
    }
}
```


다음은 출력 항목에 대한 설명입니다.

출력 항목	설명
cache_map	캐시에서 관리하는 키(URI)와 관련 정보들을 반환
cache_map/expire_time	해당 아이템이 만료되는 시각
cache_map/key	해당 아이템의 URI
cache_map/size	해당 캐시 아이템의 전체 크기
cache_map_size	캐시에서 관리 중인 캐시 아이템 건수
memory_usage	캐시 아이템 콘텐츠에 대해 실제 사용 중인 메모리 총량 (단위: byte)

5. WebtoB 콘솔 툴

WebtoB는 엔진 프로세스 및 서버 프로세스들을 관리하기 위해서 다음과 같은 툴을 제공합니다.

- **Admin 툴**

콘솔 툴	설명
wsadmin	WebtoB 시스템 전체적인 관리를 위해서 사용되는 툴로서, 시스템 정보 조회 및 관리자 작업 수행을 지원합니다.

- **기타 툴**

콘솔 툴	설명
configValidator	WebtoB 환경 파일에 대한 스키마 검증을 수행합니다.
mkpwd	SSL 인증서 키 패스워드에 대해 암호를 저장하는 파일을 생성합니다.

5.1. wsadmin

wsadmin은 텍스트 기반의 관리 환경을 제공합니다. 항상 프롬프트(prompt) 상태로 대기하면서 입력되는 명령어를 해석하여 실행합니다.

- 실행

  wsadmin 툴을 실행하려면 **wsadmin** 명령을 사용합니다.

  ```
  $ wsadmin
  ```

  wsadmin 툴이 정상적으로 실행되면 다음과 같은 메시지와 함께 프롬프트가 표시됩니다.

  ```
  $$1 [wsadmin]>>
  ```

- 종료

  wsadmin 툴을 종료하려면 **exit** 명령을 사용합니다.

  ```
  $$3 [wsadmin]>> exit
  ```

다음은 wsadmin이 제공하는 명령어에 대한 설명입니다.

명령어	약자	설명
cache-list		HTTP 응답 캐시에 저장된 응답의 정보를 출력합니다.

명령어	약자	설명
client-info	(ci)	접속 웹 브라우저를 확인합니다.
config	(cfg)	환경 설정 내용을 조회합니다.
exit		wsadmin을 종료합니다.
stat-info	(st)	프로세스, 스레드 및 서비스 상태에 대한 통계를 조회합니다.
svg-info	(si)	서버 정보를 확인합니다.

5.2. configValidator

configValidator는 WebtoB 설정 파일에 대한 스키마 검증을 수행합니다.

WebtoB를 기동하기 전에 미리 configValidator 툴을 통해 설정 파일에 대한 검증 결과를 확인할 수 있습니다.

WebtoB의 설정 파일은 기본적으로 webtob-config.json이며, 환경 변수 WEBTOB6_CONFIG_FILE_NAME을 통해 설정 파일을 변경할 수 있습니다.

- 사용법

```
$ configValidator
```

- 사용 예시
  - 정상적인 WebtoB 설정 파일 검증

```
$ configValidator
Config file path: ../config/webtob-config.json
Schema file path: ../config/webtob-config.schema.json
WEBTOB6_HOME_PATH = ../
WEBTOB6_CONFIG_FILE_PATH = ../config/
WEBTOB6_LIBRARY_PATH = ../lib/
WEBTOB6_SSL_PATH = ../ssl/
WEBTOB6_LICENSE_PATH = ../license/
WEBTOB6_SCHEMA_PATH = ../schema/
WEBTOB6_CONFIG_FILE_NAME = webtob-config.json
******* Start config validation *******
1. Opened schema file
2. Opened config file
3. Config parsing finished
4. Pre-validation process finished
5. Validation using json-schema finished
6. Post-validation process finished
Config file "../config/webtob-config.json" is validated with schema file "../schema/webtob-
config.schema.json"
Success to load config files : webtob-config.json
******* Validation success *******
```

- 비정상적인 WebtoB 설정 파일 검증

```
$ configValidator
Config file path: ../config/webtob-config.json
Schema file path: ../config/webtob-config.schema.json
WEBTOB6_HOME_PATH = ../
WEBTOB6_CONFIG_FILE_PATH = ../config/
WEBTOB6_LIBRARY_PATH = ../lib/
WEBTOB6_SSL_PATH = ../ssl/
WEBTOB6_LICENSE_PATH = ../license/
WEBTOB6_SCHEMA_PATH = ../schema/
WEBTOB6_CONFIG_FILE_NAME = webtob-config.json
******* Start config validation *******
1. Opened schema file
2. Opened config file
3. Config parsing finished
4. Pre-validation process finished
[Exception on json-schema validation][Config validation failed]At /node/hth_count of -1 -
instance is below minimum of 1
```

5.3. mkpwd

mkpwd는 SSL 절의 certificate_key_password를 지원하기 위한 툴입니다.

SSL 절에 암호화된 개인키를 설정하면 WebtoB를 기동할 때마다 암호문 입력을 요구합니다. 매번 암호문을 입력하는 번거로움을 줄이기 위해 certificate_key_password를 설정할 수 있습니다. mkpwd는 certificate_key_password에 적용할 수 있도록 passphrase 암호를 저장하는 passphrase 파일을 생성합니다.

certificate_key_password 사용법은 SSL 절 설정 항목을 참고합니다.

- 사용법

```
$ mkpwd <file_path> <ssl_name>
```

옵션	설명
<file_path>	certificate_key_password에 사용되는 파일 이름으로 mkpwd 실행의 결과를 해당 파일에 (추가)저장합니다.
<ssl_name>	SSL 절에 설정한 이름입니다.

- 사용 예시
  - 옵션 없이 실행

```
$ mkpwd
<< Usage >>
$ mkpwd file_path ssl_name
```

```
     file_path: output file for ssl certificate key password
     ssl_name: name of SSL section
```

- file_path과 ssl_name을 지정하여 실행

```
$ mkpwd ssl.ppd ssl1
Make password for SSL certificate key password
Enter password: (암호문 입력)
Successfully Added password for [ssl1] to a file [ssl.ppd].

$ls -al ssl.ppd
-rw-rw-r--  1 webtob webtob 14 Nov 6 12:34 ssl.ppd
```

- 파일 내용 확인

```
$ cat ssl.ppd
ssl1 dGVzdA==
```