



# [webMethods] HTTP - multipart/form-data

## References

- MIME Types - form-data
- multipart/form-data 란
- multipart/form-data 메시지 구성 방법

- **Multipart/form-data 란**

- Multipart 란 여러종류의 데이터를 여러부분으로 나누어 보낼 때 사용하는 Content-type
- Form 전송은 여러 종류의 데이터를 한번에 보낼 수 있음
- 파일 업로드 상황에서 한 요청의 Body에 두 종류 이상의 데이터를 구분해서 호출하는 방법이 필요하여 생긴 Content-type
  - 예를 들어 이미지(image/jpeg) + 이미지 설명(application/x-www-form-urlencoded)

- **multipart/form-data 메시지 구성 방법**

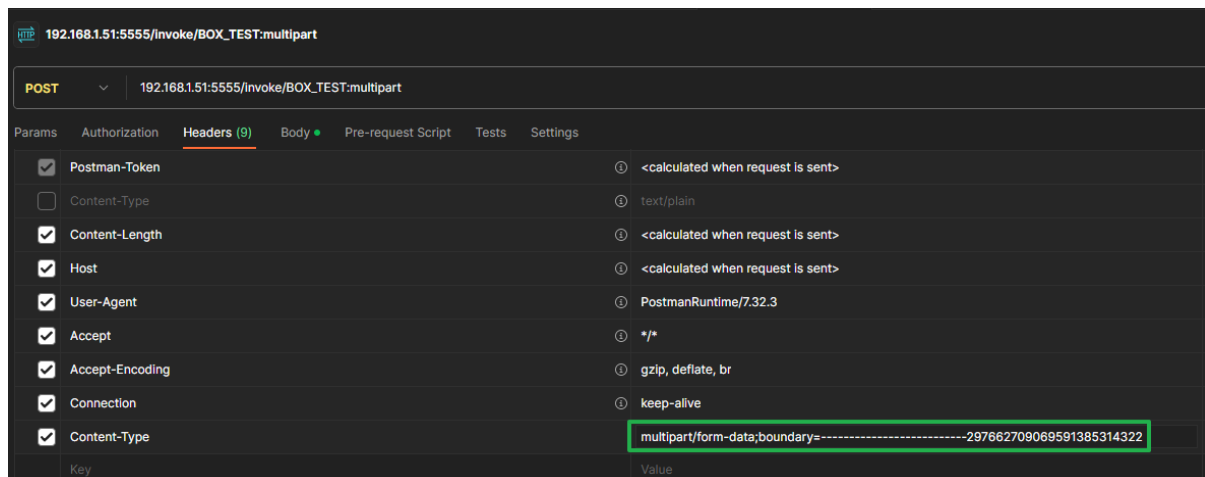
- Request Header 예 Content-type : multipart/form-data ; {Boundary 문자열} 로 구성

- Boundary 문자열은 메시지 페이로드를 각 파트, 데이터를 구분하는 구분자
- {개행} Boundary 문자열 {개행} 으로 데이터가 구분 되며 개행에는 반드시 \r\n 을 사용해야 함
- 각 데이터 구분 값은 -- 로 시작해야 하며 앞에 -- 를 붙여 구분, 모든 파트가 끝나면 구분자 뒤에 -- 를 추가하여 끝맺음
- 각 파트에는 헤더가 들어가며 Content-Disposition:form-data; name=" ..." 와 같은 폼 필드의 정보를 추가
  - Content-Disposition 기본값은 form-data
  - name : Request 필드 Key 값
  - filename : 파일 명

#### ※ POSTMAN 으로 호출한 Request Body Example (w. Wireshark)

Request body 의 form-data 기능을 사용하여 호출하면 편리하지만

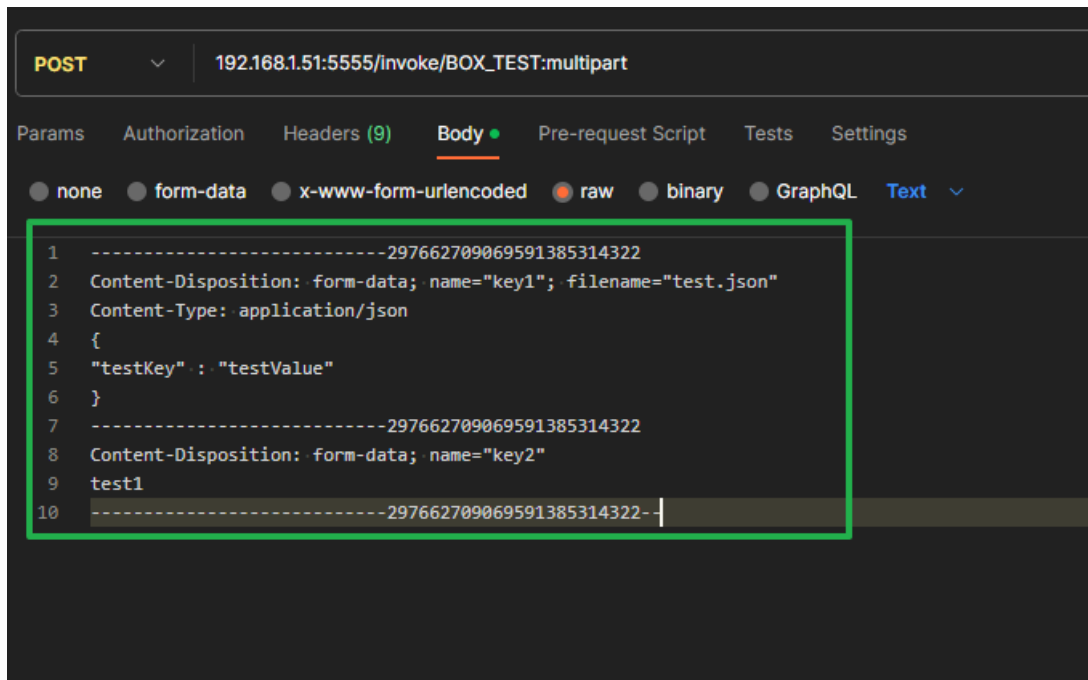
커스텀 boundary 설정 및 raw 데이터 호출을 통해 multipart/form-data 요청 상세 확인



Request Header 세팅 - 커스텀 Boundary 값

ex) multipart/form-data; boundary=-----297662709069591385314322

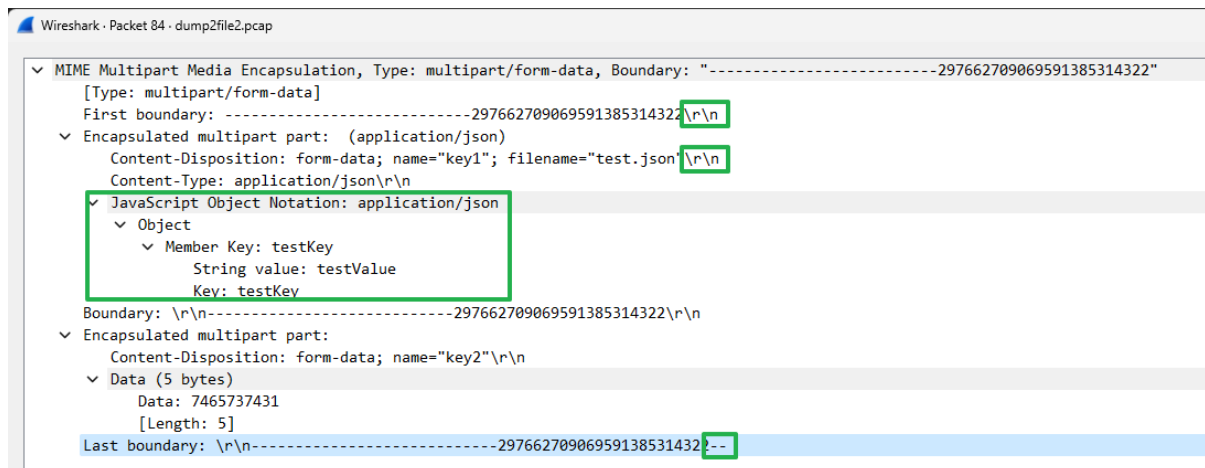
**Note.** multipart/form-data 는 boundary 값이 필요하며, 세미콜론(;)로 구분



#### Request Body 예시 - raw 사용 시

form-data 를 사용해도 되지만, raw 사용시 다음과 같이 /r/n이 제거된 상태로 호출 되어야 함.

\*주의\* "--" + "-----297662709069591385314322" (헤더에 설정된 바운더리 값에 -- 추가 됨)



#### Note.

개행은 반드시 \r\n 으로 구분

각 데이터 구분 First Boundary 값은 "--" + "헤더에 설정 한 boundary 값" 으로 설정

Last boundary 는 구분 바운더리 값에 -- 를 추가하여 마무리

("--" + "헤더에 설정 한 boundary 값" + "--")

#### [호출 상세]

```

POST /invoke/BOX_TEST:multipart HTTP/1.1
User-Agent: PostmanRuntime/7.32.3
Accept: */*
Postman-Token: a4ae8d3b-3f16-4c76-9542-80041108a45f
Host: 192.168.1.51:5555
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Content-Type: multipart/form-data; boundary=-----297662709069591385314322

Content-Length: 353

-----297662709069591385314322
Content-Disposition: form-data; name="key1"; filename="test.json"
Content-Type: application/json
{
  "testKey" : "testValue"
}
-----297662709069591385314322
Content-Disposition: form-data; name="key2"
test1
-----297662709069591385314322--

```

**[추가 예시]** 다중 이미지 파일 업로드 요청 시 서버 수신 HTTP 메시지 내용 예시  
 커스텀 Boundary 값 사용 예시 (myboundaryvalue)

```

POST /submit HTTP/1.1
Host:my.server.com
User-Agent: Mozilla/5.0 Gecko/2009042316 FireFox/3.0.10
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Referer: http://my.server.com/test/index.html
Content-type: multipart/form-data; boundary=myboudnaryvalue
Content-length: 514

--myboudnaryvalue
Content-Disposition: form-data; name="datafiled1"; filename="r.gif"
Content-Type: image/gif
GIF871.....D..;

--myboudnaryvalue
Content-Disposition: form-data; name="datafiled1"; filename="g.gif"
Content-Type: image/gif
GIF87a.....D..;

--myboudnaryvalue
Content-Disposition: form-data; name="datafiled1"; filename="b.gif"
Content-Type: image/gif
GIF87a.....D..;
--myboudnaryvalue--

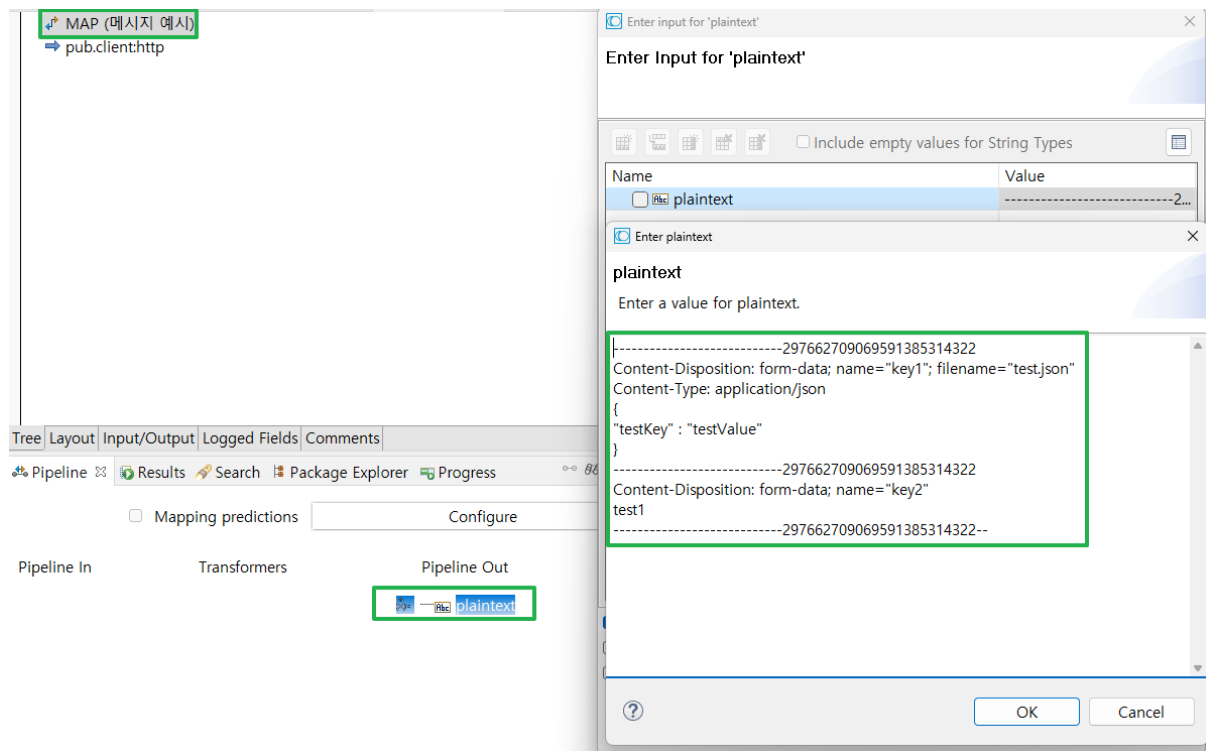
```

※ webMethods Integration Server 에서 multipart/form-data 호출 방안 (10 15 버전 기준)

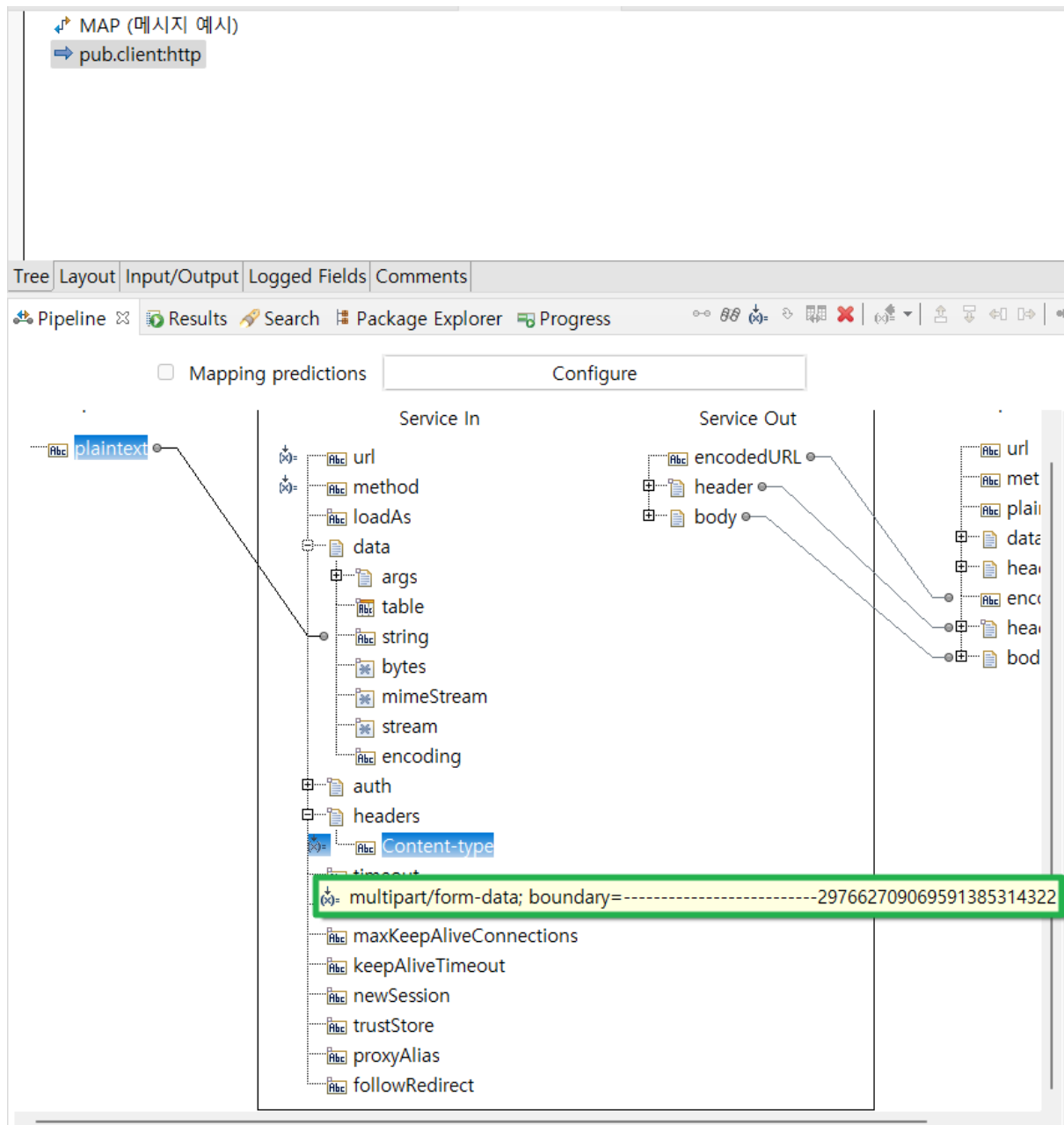
Provider로 노출 된 API 의 multipart Content-type 을 지원하지 않음 (Inbound 호출 불가)

IS 내부에서 직접 Request Body를 생성하여 Outbound 호출만 가능

Outbound 호출 예시 서비스



multipart/form-data 메시지 생성 Rule 에 맞게 Request body 생성



상세 호출 방안

## Wireshark 로 Outbound HTTP 호출 검증 시 - 정상 처리 확인

11.408851996	192.168.1.75	192.168.1.51	HTTP	616 POST /wm-message HTTP/1.1 (application/x-wmdatabin)
11.420626931	192.168.1.51	192.168.1.79	HTTP/JSON	632 POST /invoke/test:test_sjh HTTP/1.1 , JavaScript Object Notation (application/json)
11.432990996	192.168.1.79	192.168.1.51	HTTP	1623 HTTP/1.1 200 OK (text/html)
11.434351851	192.168.1.51	192.168.1.75	HTTP	373 HTTP/1.1 200 OK (text/html)

```

> Internet Protocol Version 4, Src: 192.168.1.51, Dst: 192.168.1.79
> Transmission Control Protocol, Src Port: 34342, Dst Port: 5555, Seq: 1, Ack: 1, Len: 566
> Hypertext Transfer Protocol
> MIME Multipart Media Encapsulation, Type: multipart/form-data, Boundary: "-----297662709069591385314322"
  [Type: multipart/form-data]
  First boundary: -----297662709069591385314322\r\n
  Encapsulated multipart part: (application/json)
    Content-Disposition: form-data; name="key1"; filename="test.json"\r\n
    Content-Type: application/json\r\n
    JavaScript Object Notation: application/json
      > Object
    Boundary: \r\n-----297662709069591385314322\r\n
  Encapsulated multipart part:
    Content-Disposition: form-data; name="key2"\r\n
    Data (5 bytes)
      Data: 7465737431
      [Length: 5]
    Last boundary: \r\n-----297662709069591385314322--

```