

# I. 시스템 개요

## 1. 개요

Upload API는 파일을 CDN Storage에 upload, 조회, 삭제하는 기능을 지원합니다.

항목	내용
Upload	파일을 CDN Storage에 Upload
Listup	CDN Storage에서 주어진 경로에 대한 파일, 디렉토리 조회
Delete	요청된 디렉토리, 파일을 CDN Storage 상에서 삭제함
UnZIP	Upload 된 zip파일을 압축 해제 한 후 zip 파일이 있는 같은 Directory에 저장
Copy	CDN Storage 상의 content를 복사
Move	CDN Storage 상의 content를 이동
Mkdir	CDN Storage 상에서 directory 생성
Download	CDN Storage 상의 content를 download
Upload( PUT )	CDN Storage에 content upload ( PUT method )
Rename	CDN Storage 상의 content를 rename

## 2. 지원 프로토콜

모든 데이터는 HTTP 기반으로 송수신 합니다.

- REST(HTTP POST/GET)

### 3. Client 개발 환경

---

Upload API는 Flow.js(<https://github.com/flowjs/flow.js>)라는 open source를 이용하였으며 Upload를 수행하는 Client에서는 **Nimbus에서 수정한 flow.js 파일을 이용하여 upload client를 구현하여야 합니다.**

Upload API는, 아래와 같이 일반적인 HTTP POST 방식을 통해서 이용할 수 있습니다.

#### ■ HTTP METHOD

POST/GET

#### ■ URL

`http://[UPLOADAPI-Domain-Name]/[Method Name]`

위의 URL 중에 [Method Name] 부분에는 호출할 API 명을 입력합니다. (upload, list, delete)  
[UPLOADAPI-Domain-Name]부분에는 실제 upload api의 도메인 주소와 port값을 입력합니다.  
입력 변수들은 POST/GET 방식으로 전달할 수 있습니다.

## II. 송/수신 데이터 구조

Upload API 서버는 수행 시 Client와 규정된 데이터 구조의 형식으로 송/수신 처리합니다.  
서버와 Client간의 데이터는 POST 방식을 가지며, chunked 형식으로 파일 데이터를 UPLOAD합니다.  
Upload API 서버는 OpenSource Flow.js(<https://github.com/flowjs/flow.js>)를 이용하였으며 Upload를 위한 **Client도 Nimbus에서 수정한 flow.js file을 이용하여 구현해야 합니다.**

### 1. Upload

Client는 Upload API 서버에 contents를 Upload하기 위해서 Upload API 서버에 Upload-Check를 요청하여 이상이 없는 지 확인한 후 chunk단위로 multi part upload를 시작합니다. Client에서는 Nimbus에서 수정한 Flow.js를 이용하여 upload를 수행하면 되는데 아래의 parameter가 추가되어야 합니다. 이 외에도 Upload에 필요한 여러가지 parameter가 있는데 이는 Nimbus에서 수정한 flow.js file에서 자동으로 처리됩니다.

#### 1.1 Upload 요청

##### 1.1.1 POST 방식

■ Method Name : upload

parameter	description	비고
userid	FTP user id	ex) test
passwd	FTP user id password	ex) 1234
auth_token	FTP user id auth token passwd 대신 사용 가능	ex) abc6e615-6f36-419a-b961-3a0df78f8a96
subpath	Upload할 FTP에서의 subpath	ex) testsubpath1/16/06/05
callback_url	Upload 완료 후 호출할 callback url	ex) http://test.co.kr/uploadcompleted

JSON 파일 내용은 [부록.B](#) 참조

## 1.2 Upload 완료시 Call back

Upload API 서버에 Upload를 시작하면 Client에서 data가 각각의 file마다 chunk 단위로 Upload API 서버로 Upload 됩니다. Upload 서버에서는 각각의 chunk를 저장해놨다가 모든 Chunk 전송이 완료 되면, 각각 본래의 원본 파일로 합칩니다. 이 과정이 모두 끝나면 Upload시 지정했던 callback\_url을 HTTP POST method로 호출하여 Upload가 완료된 파일의 정보를 Json 형식으로 전달합니다.

parameter	description	비고
request_id	해당 파일의 upload request id	ex) 42a67530-3752-11e6-bfdd-71b7f85d5fc7
service	Upload Service ID( 고정 )	ex) NIM10001
title	Upload file Name	ex) 1.mp4

JSON 파일 내용은 [부록.C](#) 참조

## 2. CDN Storage 조회

CDN Storage에 존재하는 파일의 정보를 제공하는 기능을 제공합니다.

주어진 경로가 Directory인 경우 해당 directory 내의 directory와 file들의 정보를 제공합니다.

주어진 경로가 file인 경우 해당 file의 정보만을 제공합니다.

주어진 경로가 존재하지 않는 경우 에러 응답을 합니다.

### 2.1 List up 요청

#### 2.1.1 POST 방식

- Method Name : listup

**주의 :** Client에서 Request 생성 시, http header에 “Content-Type: Application/json”을 반드시 추가해야 합니다.

parameter	description	비고
userid	FTP user id	ex) test
passwd	FTP user id password	ex) 1234
auth_token	FTP user id auth token passwd 대신 사용 가능	ex) abc6e615-6f36-419a-b961-3a0df78f8a96
path	조회할 FTP에서의 path	ex) testsubpath1/16/06/05 혹은 testsubpath55/15/07/23/2.mp4

JSON 파일 내용은 [부록.D](#) 참조

## 2.2 List Up 응답

parameter		description	비고
result_code		Result code	ex) 200, 401, 404, 500
result_msg		Result message	Ex) success, not found
path		조회 대상이 된 path	ex) testsubpath1/16/06/05 혹은 testsubpath55/15/07/23/2.mp4
list	name	File or Directory name	ex) 1.mp4
	type	Directory, File Type	ex) F : file, D : directory
	last-modified	가장 최근에 수정(생성)된 Date(yyyyMMddHHmmss)	ex) 20160602153824
	size	File Size	ex) 110011072

위 항목에서 list 항목은 json array로 구현됩니다.  
JSON 파일 내용은 [부록.E](#) 참조

## 3. CDN Storage에서의 경로 삭제

CDN Storage에 존재하는 파일을 삭제하는 기능을 제공합니다.

주어진 경로가 Directory인 경우 해당 directory 내의 directory와 file들을 모두 삭제하는 기능을 제공합니다.

**삭제는 매우 위험하니 신중하게 처리해야 합니다.**

### 3.1 Delete 요청

#### 3.1.1 POST 방식

- Method Name : delete

**주의 : Client에서 Request 생성 시, http header에 “Content-Type: Application/json”을 반드시 추가해야 합니다.**

parameter	description	비고
userid	FTP user id	ex) test
passwd	FTP user id password	ex) 1234
auth_token	FTP user id auth token passwd 대신 사용 가능	ex) abc6e615-6f36-419a-b961-3a0df78f8a96
path	삭제할 FTP에서의 path	ex) testsubpath1/16/06/05 혹은 testsubpath55/15/07/23/2.mp4

JSON 파일 내용은 [부록.F](#) 참조

### 3.2 Delete 응답

parameter	description	비고
result_code	Result code	ex) 200, 401, 404, 500
result_msg	Result message	ex) success, not found
path	조회 대상이 된 path	ex) testsubpath1/16/06/05 혹은 testsubpath55/15/07/23/2.mp4

JSON 파일 내용은 [부록.G](#) 참조

## 4. UnZIP

Upload API 서버에 Upload 되어 있는 파일 중 \*.zip 형식의 파일의 압축을 풀어서 해당 zip 파일과 같은 경로상에 위치하게 하는 기능을 제공합니다.

**주의 : Client에서 Request 생성 시, http header에 “Content-Type: Application/json”을 반드시 추가해야 합니다.**

**주의 : UnZIP 매소드에서는 한번에 하나의 zip 파일 만을 압축 해제 합니다.**

### 4.1 UnZIP 요청

#### 4.1.1 POST 방식

- Method Name : extractzip

parameter	description	비고
userid	FTP user id	ex) test
passwd	FTP user id password	ex) 1234
auth_token	FTP user id auth token passwd 대신 사용 가능	ex) abc6e615-6f36-419a-b961-3a0df78f8a96
path	UnZIP 할 FTP에서의 파일 경로	ex) testsubpath1/16/06/05.zip

JSON 파일 내용은 [부록.H](#) 참조

### 4.2 UnZIP 응답

Upload API 서버에 UnZIP 에 대한 요청을 하면 해당 path에 UnZIP 이 시작이 되고 해당 UnZIP 에 대한 작업이 끝나면 해당 path 에 대한 정보를 Json 형식으로 전달합니다.

parameter	description	비고
result_code	Result code	ex) 200, 400, 401, 500
result_msg	Result message	ex) unzip success, zip file path is not set, invalid_auth_info, error in unzip content
path	UnZIP 대상이 된 path	ex) testsubpath1/16/06/05.zip

JSON 파일 내용은 [부록.I](#) 참조



## 5. Copy

Upload API 서버에 Upload 되어 있는 file 이나 directory를 주어진 경로로 복사하는 기능을 제공합니다.

**주의 : Client에서 Request 생성 시, http header에 “Content-Type: Application/json”을 반드시 추가해야 합니다.**

**주의 : Copy 매소드에서는 한번에 하나의 file 혹은 directory만을 복사합니다.**

### 5.1 Copy 요청

#### 5.1.1 POST 방식

■ Method Name : copy

parameter	description	비고
userid	FTP user id	ex) test
passwd	FTP user id password	ex) 1234
auth_token	FTP user id auth token passwd 대신 사용 가능	ex) abc6e615-6f36-419a-b961-3a0df78f8a96
src	복사할 원본 file 혹은 directory list	ex) ["testsubpath1/16/06/05.txt", "testsubpath1/16/06"]
dest	복사할 목적지 경로	ex) destdir/path/directory

JSON 파일 내용은 [부록.J](#) 참조

### 5.2 Copy 응답

Upload API 서버에 Copy 에 대한 요청을 하면 원본 file 혹은 directory를 목적지 경로로 복사를 하고 복사가 완료되면 원본 및 목적지 경로에 대한 정보를 Json 형식으로 전달합니다.

parameter	description	비고
result_code	Result code	ex) 200, 400, 401, 500
result_msg	Result message	ex) copy success, error in copying content, invalid_auth_info, source or destination is not set
src	복사할 원본 file 혹은 directory list	ex) ["testsubpath1/16/06/05.txt", "testsubpath1/16/06"]

parameter	description	비고
dest	복사할 목적지 경로	ex) destdir/path/directory

JSON 파일 내용은 [부록.K](#) 참조

## 6. Move

Upload API 서버에 Upload 되어 있는 file 이나 directory를 주어진 경로로 이동하는 기능을 제공합니다.

**주의 : Client에서 Request 생성 시, http header에 “Content-Type: Application/json”을 반드시 추가해야 합니다.**

**주의 : Move 매소드에서는 한번에 하나의 file 혹은 directory만을 이동합니다.**

### 6.1 Move 요청

#### 6.1.1 POST 방식

■ Method Name : move

parameter	description	비고
userid	FTP user id	ex) test
passwd	FTP user id password	ex) 1234
auth_token	FTP user id auth token passwd 대신 사용 가능	ex) abc6e615-6f36-419a-b961-3a0df78f8a96
src	이동할 원본 file 혹은 directory list	ex) ["testsubpath1/16/06/05.txt", "testsubpath1/16/06"]
dest	이동할 목적지 경로	ex) destdir/path/directory

JSON 파일 내용은 [부록.L](#) 참조

### 6.2 Move 응답

Upload API 서버에 Move에 대한 요청을 하면 원본 file 혹은 directory를 목적지 경로로 이동하고 이동이 완료 되면 원본 및 목적지 경로에 대한 정보를 Json 형식으로 전달합니다.

parameter	description	비고
result_code	Result code	ex) 200, 400, 401, 500
result_msg	Result message	ex) move success, error in copying content, invalid_auth_info, source or destination is not set
src	이동할 원본 file 혹은 directory list	ex) ["testsubpath1/16/06/05.txt", "testsubpath1/16/06"]

parameter	description	비고
dest	이동할 목적지 경로	ex) destdir/path/directory

JSON 파일 내용은 [부록.M](#) 참조

## 7. Mkdir

Upload API 서버에 Directory를 생성하는 기능을 제공합니다. 만약 생성 경로에 포함되는 중간 directory가 생성되어 있지 않으면 모두 생성합니다.

**주의 : Client에서 Request 생성 시, http header에 “Content-Type: Application/json”을 반드시 추가해야 합니다.**

**주의 : Mkdir 매소드에서는 한번에 하나의 directory만을 생성합니다.**

### 7.1 Mkdir 요청

#### 7.1.1 POST 방식

- Method Name : mkdir

parameter	description	비고
userid	FTP user id	ex) test
passwd	FTP user id password	ex) 1234
auth_token	FTP user id auth token passwd 대신 사용 가능	ex) abc6e615-6f36-419a-b961-3a0df78f8a96
path	생성해야 할 directory 경로	ex) testsubpath1/16/06

JSON 파일 내용은 [부록.N](#) 참조

### 7.2 Mkdir 응답

Upload API 서버에 mkdir에 대한 요청을 하면 해당 경로에 대응하는 directory를 생성하고 생성이 완료되면 생성된 directory 경로에 대한 정보를 Json 형식으로 전달합니다.

parameter	description	비고
result_code	Result code	ex) 200, 400, 401, 500
result_msg	Result message	ex) success, invalid_auth_info
path	생성해야 할 directory 경로	ex) testsubpath1/16/06

JSON 파일 내용은 [부록.Q](#) 참조

## 8. Download

Upload API 서버에 있는 content를 http protocol로 다운받는 기능을 제공합니다.

**주의 : GET 방식을 사용하며 http header에 “Content-Type: Application/json”을 추가하지 않습니다.**  
**client에서 request url 생성 시 query parameter로 userid, passwd, path을 설정해야 하며 모든 항목은 url encoding을 수행한 값으로 설정되어야 합니다.**

### 8.1 Download 요청

#### 8.1.1 GET 방식

- Method Name : download

parameter	description	비고
userid	FTP user id	ex) test
passwd	FTP user id password	ex) 1234
auth_token	FTP user id auth token passwd 대신 사용 가능	ex) abc6e615-6f36-419a-b961-3a0df78f8a96
path	다운 받을 content file 경로	ex) testsubpath1/16/06/05.txt ( url encoding : testsubpath1%2F16%2F06%2F05.txt )

JSON 파일 내용은 [부록.P](#) 참조

### 8.2 Download 응답

Upload API 서버에 download에 대한 요청을 하면 해당 경로에 대응하는 content의 data를 http protocol을 통해 client로 보내 줍니다. 아래는 실패한 경우에 대한 코드 및 응답 정의입니다.

parameter	description	비고
result_code	Result code	ex) 400, 401, 403, 404, 500
result_msg	Result message	ex) does not exists, Directory could not be downloaded, invalid_auth_info, internal server error
path	다운 받을 content file 경로	ex) testsubpath1/16/06/05.txt

JSON 파일 내용은 [부록.Q](#) 참조

## 9. Upload( PUT )

Upload API는 Client에서 local file을 서버로 http push method를 이용하여 upload하는 기능을 제공합니다.

**주의 : PUT 방식을 사용하며 http header에 “Content-Type:application/octet-stream”과 “Content-Length:sizeValue”를 반드시 설정하여야 합니다.**

**client에서 request url 생성 시 query parameter로 userid, passwd, path을 설정해야 하며 모든 항목은 url encoding을 수행한 값으로 설정되어야 합니다.**

### 9.1 Upload( PUT ) 요청

#### 9.1.1 PUT 방식

- Method Name : uploadput

parameter	description	비고
userid	FTP user id	ex) test
passwd	FTP user id password	ex) 1234
auth_token	FTP user id auth token passwd 대신 사용 가능	ex) abc6e615-6f36-419a-b961-3a0df78f8a96
subpath	Upload할 콘텐츠가 저장 될 directory 경로	ex) testsubpath1/16/06 ( url encoding : testsubpath1%2F16%2F06 )
filename	Upload 할 콘텐츠가 저장 될 file name	ex) 05.txt

JSON 파일 내용은 [부록.R](#) 참조

## 9.2 Upload( PUT ) 응답

Upload API 서버에 upload( PUT )에 대한 요청을 하면 upload api에서는 upload를 수행하고 완료된 후 응답을 client로 보냅니다.

parameter	description	비고
result_code	Result code	ex) 200, 400, 401, 403, 500
result_msg	Result message	ex) success, file name isn't defined, Directory( its name is same with file name ) exists on remote file path, invalid_auth_info, error in writing file, error in renaming file, internal server error
subpath	Upload할 콘텐츠가 저장 될 directory 경로	ex) testsubpath1/16/06 ( url encoding : testsubpath1%2F16%2F06 )
filename	Upload 할 콘텐츠가 저장 될 file name	ex) 05.txt

JSON 파일 내용은 [부록.S](#) 참조



## 10. Rename

Upload API 서버에 Upload 되어 있는 file 이나 directory를 주어진 경로로 rename하는 기능을 제공합니다.

**주의 : Client에서 Request 생성 시, http header에 “Content-Type: Application/json”을 반드시 추가해야 합니다.**

**주의 : Rename 매소드에서는 한번에 하나의 file 혹은 directory만을 rename합니다.**

### 10.1 Rename 요청

#### 10.1.1 POST 방식

- Method Name : rename

parameter	description	비고
userid	FTP user id	ex) test
passwd	FTP user id password	ex) 1234
auth_token	FTP user id auth token passwd 대신 사용 가능	ex) abc6e615-6f36-419a-b961-3a0df78f8a96
src	rename할 원본 file 혹은 directory	ex) testsubpath1/16/06/05.txt
dest	rename할 목적지 경로	ex) destdir/path/directory/dest

JSON 파일 내용은 [부록.T](#) 참조

### 10.2 Rename 응답

Upload API 서버에 Rename에 대한 요청을 하면 원본 file 혹은 directory를 목적지 경로로 rename하고 완료 되면 원본 및 목적지 경로에 대한 정보를 Json 형식으로 전달합니다.

parameter	description	비고
result_code	Result code	ex) 200, 400, 401, 500
result_msg	Result message	ex) rename success, error in renaming content, invalid_auth_info, source or destination is not set
src	rename할 원본 file 혹은 directory	ex) testsubpath1/16/06/05.txt

parameter	description	비고
dest	rename할 목적지 경로	ex) destdir/path/directory/dest

JSON 파일 내용은 [부록.U](#) 참조

# 부록 A. 응답(에러) 코드

코드	내역	비고
200	성공	
200	요청 성공	요청에 대한 처리 완료
4XX	실패	
400	Parameter 입력 오류	Parameter 입력이 누락되었거나 인증 안됨
401	ID/Password 입력 오류	ID/Password 입력이 누락되었거나 인증 안됨
404	경로가 존재하지 않음	CDN Storage 상에 해당 경로가 존재하지 않음
5XX	오류	
500	내부 오류	

## 부록 B. UPLOAD 요청

---

Upload API 서버의 upload 요청 내용은 아래와 같습니다.

```
{
  "userid": "test",
  "passwd": "1234",
  "subpath": "testsubpath1/16/06/05",
  "callback_url": "http://test.co.kr/uploadcompleted"
}
```

혹은 passwd 대신에 auth\_token 설정 시

```
{
  "userid": "test",
  "auth_token": "abc6e615-6f36-419a-b961-3a0df78f8a96",
  "subpath": "testsubpath1/16/06/05",
  "callback_url": "http://test.co.kr/uploadcompleted"
}
```

## 부록 C. UPLOAD 완료 CALL BACK

---

```
{
  "request_id": "42a67530-3752-11e6-bfdd-71b7f85d5fc7",
  "service": "NIM10001",
  "title": "1.mp4"
}
```

## 부록 D. LISTUP 요청

---

Directory 조회를 요청한 경우

```
{
  "userid": "test",
  "passwd": "1234",
  "path": "testsubpath1/16/06/05"
}
```

혹은 passwd 대신에 auth\_token 설정 시

```
{
  "userid": "test",
  "auth_token": "abc6e615-6f36-419a-b961-3a0df78f8a96",
  "path": "testsubpath1/16/06/05"
}
```

File 조회를 요청한 경우

```
{
  "userid": "test",
  "passwd": "1234",
  "path": "testsubpath1/16/06/05/1.mp4"
}
```

혹은 passwd 대신에 auth\_token 설정 시

```
{
  "userid": "test",
  "auth_token": "abc6e615-6f36-419a-b961-3a0df78f8a96",
  "path": "testsubpath1/16/06/05/1.mp4"
}
```

## 부록 E. LISTUP 응답

---

Directory 조회를 요청한 경우

```
{
  "result_code": "200",
  "result_msg": "success",
  "path": "testsubpath1/16/06/05",
  "list": [
    {
      "name": "1.mp4",
      "type": "F",
      "last-modified": "20160602153824",
      "size": "110011072"
    },
    {
      "name": "2.mp4",
      "type": "F",
      "last-modified": "20160610231219",
      "size": "265938726"
    },
    {
      "name": "12",
      "type": "D",
      "last-modified": "20160617101135",
      "size": "4096"
    }
  ]
}
```

File 조회를 요청한 경우

```
{
  "result_code": "200",
  "result_msg": "success",
  "path": "testsubpath1/16/06/05/1.mp4",
  "list": [
    {
      "name": "1.mp4",
      "type": "F",
      "last-modified": "20160602153824",
      "size": "110011072"
    }
  ]
}
```

## 부록 F. DELETE 요청

---

Directory 삭제를 요청한 경우

```
{  
  "userid": "test",  
  "passwd": "1234",  
  "path": "testsubpath1/16/06/05"  
}
```

혹은 passwd 대신에 auth\_token 설정 시

```
{  
  "userid": "test",  
  "auth_token": "abc6e615-6f36-419a-b961-3a0df78f8a96",  
  "path": "testsubpath1/16/06/05"  
}
```

File 삭제를 요청한 경우

```
{  
  "userid": "test",  
  "passwd": "1234",  
  "path": "testsubpath1/16/06/05/1.mp4"  
}
```

혹은 passwd 대신에 auth\_token 설정 시

```
{  
  "userid": "test",  
  "auth_token": "abc6e615-6f36-419a-b961-3a0df78f8a96",  
  "path": "testsubpath1/16/06/05/1.mp4"  
}
```

## 부록 G. DELETE 응답

---

Directory를 삭제 요청한 경우

```
{  
  "result_code": "200",  
  "result_msg": "success",  
  "path": "testsubpath1/16/06/05"  
}
```

File을 삭제 요청한 경우

```
{  
  "result_code": "200",  
  "result_msg": "success",  
  "path": "testsubpath1/16/06/05/1.mp4"  
}
```



## 부록 H. UNZIP 요청

---

File UnZIP을 요청한 경우

```
{
  "userid": "test",
  "passwd": "1234",
  "path": "testsubpath1/16/06/05.zip"
}
```

혹은 passwd 대신에 auth\_token 설정 시

```
{
  "userid": "test",
  "auth_token": "abc6e615-6f36-419a-b961-3a0df78f8a96",
  "path": "testsubpath1/16/06/05.zip"
}
```

## 부록 I. UNZIP 응답

---

File UnZIP을 요청한 경우

```
{
  "result_code": "200",
  "result_msg": "unzip success",
  "path": "testsubpath1/16/06/05.zip"
}
```

## 부록 J. COPY 요청

---

```
{
  "userid": "test",
  "passwd": "1234",
  "src": [
    "testsubpath1/16/06/05.txt",
    "testsubpath1/16/06"
  ],
  "dest": "destdir/path/directory"
}
```

혹은 passwd 대신에 auth\_token 설정 시

```
{
  "userid": "test",
  "auth_token": "abc6e615-6f36-419a-b961-3a0df78f8a96",
  "src": [
    "testsubpath1/16/06/05.txt",
    "testsubpath1/16/06"
  ],
  "dest": "destdir/path/directory"
}
```

## 부록 K. COPY 응답

---

```
{
  "result_code": "200",
  "result_msg": "copy success",
  "src": [
    "testsubpath1/16/06/05.txt",
    "testsubpath1/16/06"
  ],
  "dest": "destdir/path/directory"
}
```

## 부록 L. MOVE 요청

---

```
{
  "userid": "test",
  "passwd": "1234",
  "src": [
    "testsubpath1/16/06/05.txt",
    "testsubpath1/16/06"
  ],
  "dest": "destdir/path/directory"
}
```

혹은 passwd 대신에 auth\_token 설정 시

```
{
  "userid": "test",
  "auth_token": "abc6e615-6f36-419a-b961-3a0df78f8a96",
  "src": [
    "testsubpath1/16/06/05.txt",
    "testsubpath1/16/06"
  ],
  "dest": "destdir/path/directory"
}
```

## 부록 M. MOVE 응답

---

```
{
  "result_code": "200",
  "result_msg": "move success",
  "src": [
    "testsubpath1/16/06/05.txt",
    "testsubpath1/16/06"
  ],
  "dest": "destdir/path/directory"
}
```

## 부록 N. MKDIR 요청

---

```
{
  "userid": "test",
  "passwd": "1234",
  "path": "testsubpath1/16/06"
}
```

혹은 passwd 대신에 auth\_token 설정 시

```
{
  "userid": "test",
  "auth_token": "abc6e615-6f36-419a-b961-3a0df78f8a96",
  "path": "testsubpath1/16/06"
}
```

## 부록 O. MKDIR 응답

---

```
{
  "result_code": "200",
  "result_msg": "mkdir success",
  "path": "testsubpath1/16/06"
}
```

## 부록 P. DOWNLOAD 요청( URL )

---

```
http://example.com:3800/download?userid=test&passwd=1234&path=testsubpath1%2F16%2F06%2F05.txt
```

혹은 passwd 대신에 auth\_token 설정 시

```
http://example.com:3800/download?userid=test&auth_token=abc6e615-6f36-419a-b961-3a0df78f8a96&path=testsubpath1%2F16%2F06%2F05.txt
```

## 부록 Q. DOWNLOAD 응답( 오류 시 )

---

```
{  
  "result_code": "404",  
  "result_msg": "does not exists",  
  "path": "testsubpath1/16/06/05.txt"  
}
```

## 부록 R. UPLOAD( PUT ) 요청( URL )

---

```
http://example.com:3800/uploadput?userid=test&passwd=1234&subpath=testsubpath1%2F16%2F06&filename=05.txt
```

혹은 passwd 대신에 auth\_token 설정 시

```
http://example.com:3800/uploadput?userid=test&auth_token=abc6e615-6f36-419a-b961-3a0df78f8a96&subpath=testsubpath1%2F16%2F06&filename=05.txt
```

## 부록 S. UPLOAD( PUT ) 응답

---

```
{
  "result_code": "200",
  "result_msg": "success",
  "subpath": "testsubpath1/16/06",
  "filename": "05.txt"
}
```

## 부록 T. RENAME 요청

---

```
{
  "userid": "test",
  "passwd": "1234",
  "src": "testsubpath1/16/06/05.txt",
  "dest": "destdir/path/directory/dest"
}
```

혹은 passwd 대신에 auth\_token 설정 시

```
{
  "userid": "test",
  "auth_token": "abc6e615-6f36-419a-b961-3a0df78f8a96",
  "src": "testsubpath1/16/06/05.txt",
  "dest": "destdir/path/directory/dest"
}
```

## 부록 U. RENAME 응답

---

```
{
  "result_code": "200",
  "result_msg": "rename success",
  "src": "testsubpath1/16/06/05.txt",
  "dest": "destdir/path/directory/dest"
}
```