청기와랩 SMS Open API Documentation

출시 1.0

BlueHouseLab DevTeam

11월 06, 2017

Contents

Contents:

CHAPTER 1

서비스 개요

청기와랩의 SMS Open API는 중소규모 웹사이트에 필요한 SMS (장문/단문) 발송 기능 구현을 오픈API 제공 서비스 입니다. 이를 위해 청기와랩의 SMS OpenAPI 서비스는 RESTful하게 디자인 되었고 JSON을 메인 전송 포멧으로 채택한 HTTP 기반 인터페이스를 제공합니다.

1.1 용어 정리

- 청기와랩 : https://bluehouselab.com
- Server : 청기와랩이 SMS OpenAPI를 위해 제공하는 WebServer
- Client : SMS OpenAPI를 사용하여 구현된 어플리케이션, 모바일앱, 웹사이트
- HTTP : Hypertext 전송 프로토콜 http://ko.wikipedia.org/wiki/HTTP
- HTTPS : HTTP over SSL, http://ko.wikipedia.org/wiki/HTTPS
- HTTP Request Header : 클라이언트가 HTTP Request시 부여할 수 있는 설정값, HTTP Request Data의 Mime Type, API Key 등이 기술 될 수 있다.
- HTTP Request Data : 클라이언트가 HTTP Request시 실어 보내는 데이타
- HTTP Response Header : Server가 클라이언트의 HTTP Request에 대해 응답할때의 각종 설정 값. HTTP Status Code (200, 204, 404, 500 ...) 및 Response Data의 Mime Type등이 기술 될 수 있다. 및 Response Data의 Mime Type등이 기술 될 수 있다.
- HTTP Response Data : Server가 클라이언트의 HTTP Request에 대해 응답 하는 데이타
- JSON : Data 교환을 위한 포멧 http://www.json.org/json-ko.html
- HTTP Basic Auth : HTTP에서 가장 많이 쓰이는 인증 방법 중의 하나 http://tools.ietf.org/html/rfc2617
- App Id : 등록된 App 또는 웹사이트의 ID
- API Key : 서비스 과금을 위해 클라이언트에게 제공되는 암호화된 문자열, App Id별로 하나씩 발급 됨
- Git : 소스코드관리도구(SCM) http://git-scm.com

- EUC-KR : KSC5601 한국어 문자세트를 표기하기 위해 유닉스에서 사용하던 문자열 인코딩 표준. 과거 인터넷 및 각종 OS에서 한국어 문자열을 표현하기 위해 널리 사용 되었으나 현재는 UTF-8, 16등의 유니코드용 인코딩 표준에 의해 대체 되었다. 한국의 SMS 망 표준으로도 사용 되었으며, 한글은 2바이트, 영문 및 특수문자는 1바이트를 차지 한다.
- UTF-8 : 유니코드 문자세트를 표기하기 위한 문자열 인코딩 표준. 한국어 뿐만 아니라 유니코드에서 표현할 수 있는 전세계 문자를 모두 인코딩 할수 있어, 각종 OS및 인터넷 관련 표준으로 사용됨. 영문 및 ASCII 영역 특수문자는 1바이트 그외 문자열은 3~6바이트로 표현 된다. 한글 문자의 경우 대부분 3바이트로 표현 된다.

1.2 Server 정보

안전한 Open API 접근을 위해 청기와랩은 HTTPS (TLSv1) 연결을 제공합니다.

- Server URL: https://api.bluehouselab.com (Port 443)

주석: API Key 보호를 위해 HTTP 연결은 제공되지 않고 HTTPS 연결만 제공 합니다. 청기와랩은 HTTPS 연결을 위해 공인 인증기관(CA)으로 부터 발급된 SSL 인증서를 제공하므로 HTTPS 연결을 믿고 쓸 수 있습니다.

만약 HTTPS 연결시 SSL 인증서 검증(Verify) 과정을 무시하도록 클라이언트를 구현할 경우 보안 공격에 의해 API Key가 노출 될 수 있으므로 주의가 필요 합니다.

1.3 App Id와 API Key

App Id는 청기와랩 SMS Open API 서비스에 등록된 App 또는 웹사이트의 Id이며 각 App Id별로 고유의 API Key가 존재 합니다. App Id와 API Key는 인증을 위해 사용 됩니다.

청기와랩 https://bluehouselab.com/sms 페이지를 방문하시면 App Id및 API Key를 발급 받으실 수 있습니다.

CHAPTER 2

RESTful API

2.1 HTTP Common Header

클라이언트는 HTTP Request 시 Header에 아래 표의 Key와 Value를 설정해야 합니다..

예제:

Key	Value	비고
Authorization	Basic Zm9vYmFyLXdlYnNpdGU6MTIzNDU2Nzg=	필수

주석: Authorization에 들어가는 HTTP Basic Auth (RFC-2617) 값은 *Authorization* 를 참조하십시오.

주석: Authorization 정보가 없으면 HTTP Status Code로 401 Authorization Required 에러를 리턴 받습니다.

2.2 Authorization

청기와랩의 SMS Open API는 HTTP 표준 Basic Auth 을 사용하여 인증 합니다. HTTP Basic Auth는 Username과 Password를 조합한 후 Base64로 인코딩한 값을 HTTP Header에 설정하여 보냅니다. 여기서는 App Id가 Username으로 사용되며 API Key가 Password로 사용됩니다.

가령 App Id가 foobar-website 이고 API Key가 12345678 이라고 하면 “foobar-website:12345678”을 Base64로 인코딩한 값을 사용하여 HTTP Header에 설정 하면 됩니다.

```
인증문자열 (Base64 Encoding 전)
Basic foobar-website:12345678

Base64 적용 후 인증 문자열
Basic Zm9vYmFyLXdlYnNpdGU6MTIzNDU2Nzg=
```

```
HTTP Header 에 들어간 완전한 인증 문자열
Authorization: Basic Zm9vYmFyLXdlYnNpdGU6MTIzNDU2Nzg=
```

2.3 HTTP Status Code

클라이언트가 API서버로 부터 받을 수 있는 HTTP Status Code 목록

Status Code	Description
200	OK, 성공
204	No Content, 성공 (Response Data 가 없는 경우)
400	Bad Request, 입력 데이타 잘못된 경우
401	Unauthorized, API Key 인증 실패
402	Payment Required, Credit 또는 Point 부족, Credit 충전 필요
403	Forbidden, 접근 불가, HTTPS를 통한 연결이 아니거나 인증 실패
404	Not found, delivery id 등의 resource가 존재하지 않는 경우
406	Not Acceptable, JSON 파싱 실패 등
410	Gone, API deprecated
412	Precondition Failed, 입력 값에 문제 있음
500	Internal Server Error, 내부 서버 오류
503	Service Unavailable, 내부 서버 오류

2.4 메시지 발송 (SMS)

HTTP Request Header:

HTTP Header	Value
Method	Post
URI Path	/smscenter/v1.0/sendsms
Content-Type	application/json; charset=utf-8

Request Data Format (JSON):

Key	Value	Format	비고
sender	“0212341234”	String	필수
receivers	[”01011111111”, “01011112222”,]	List	필수
content	“주문하신 제품이 발송 되었습니다.”	String (UTF-8)	필수
reservation	“2014-06-12T16:01:45.923Z”	UTC Date (ISO8601)	옵션
blockrejectednumbers	true	Boolean	옵션 (기본값 true)

주석: 한국의 SMS망 표준에 의거하여 EUC-KR 인코딩 기준으로 80바이트 길이 제한이 있습니다. EUC-KR 인코딩 기준 문자열 길이는 ASCII는 1바이트, 한글 및 특수 문자는 2바이트로 계산 됩니다. EUC-KR 인코딩 기준 80바이트가 넘어갈 경우 자동으로 문자열이 80바이트에 맞춰 잘라져서 발송 됩니다. 장문의 메시지를 보내고자 할경우 LMS (장문 문자)로 발송 하십시오.

주석: 청기와랩의 Open API는 ECMA-262 표준에 따라 JSON 표기시 Unicode 인코딩 (uHHHH) 또는 UTF-8인코딩으로 문자열을 처리하는 것이 기본 원칙 입니다. UTF-8 인코딩의 경우 한 글자가 1~6바이트 까지 차지할 수 있는 가변길이 문자열 인코딩 이나 Server에서 SMS 문자열 계산시에는 EUC-KR로 변환 한 후 문자열 길이를 체크 하여 발송 합니다.

주석: EUC-KR 인코딩에서 지원하지 않는 문자의 경우 '?'로 대치 됩니다. 예) '똠', '뷁'

주석: 예약 전송 옵션 사용시 reservation 값으로 JavaScript의 Date 표기법인 ISO 8601 형식의 UTC기준 시간을 기술 하십시오. 과거 또는 1분 미만의 미래 시간은 예약으로 간주되지 않습니다.

주석: 발송후 등록하신 callback URL로 발송 결과를 보내줍니다. **'발송 결과 Callback'_** 을 참고하세요.

주석: blockrejectednumbers은 기본값이 true이고 Optional한 인자 입니다. 발송시 080 수신자 부담 수신거부 시스템에 등록되어 있는 번호이면 SMS가 발송되지 않습니다. 만약 blockrejectednumbers를 false로 설정하여 발송 시 080 수신자 부담 수신거부 시스템에 등록된 전화번호를 체크하지 않고 그대로 발송 되므로 주의하여 사용하세요.

주석: 수신자가 2명 이상일 경우 groupid가 같이 리턴됩니다. groupid를 이용하여 전송 결과를 한꺼번에 받아 오실 수 있습니다.

Example:

```
{sender: "01011112222", receivers: ["01011111111",], content: "주문 취소건 발생"}
```

HTTP Response Header:

HTTP Header	Value
Status	200
Content-Type	application/json; charset=utf-8

Response Data Format (JSON):

Key	Value	Format	비고
sent	[["01011111111", "10001"],]	List	[수신번호,발송ID]의 리스트
filtered	["01011111113",]	List	수신 거부 번호들
reserved	187c4b7affa011e3bf47c42c032b413b	String	예약ID
groupid	09e9a0680ea111e49044c42c032b413b	String (Optional)	그룹ID

Example:

```
{result: [["01011111111", 20001], ["01011111112", 20002], ], filtered: [], reserved:␣
↪null}
```

2.5 메시지 발송 (LMS)

HTTP Request Header:

HTTP Header	Value
Method	Post
URI Path	/smscenter/v1.0/sendlms
Content-Type	application/json; charset=utf-8

Request Data Format (JSON):

Key	Value	Format	비고
sender	“0212341234”	String	필수
receivers	[”01011111111”, “01011112222”,]	List	필수
subject	“이벤트 알림”	String (UTF-8)	필수
content	“주문하신 제품이 발송 되었습니다.”	String (UTF-8)	필수
reservation	“2014-06-12T16:01:45.923Z”	UTC Date (ISO8601)	옵션
blockrejectednumbers	true	Boolean	옵션 (기본값 true)

Example:

```
{sender "01011112222", receivers: ["01011111111",], subject: "관리자 알림", content: "주문 취소건 발생"}
```

주석: Subject는 EUC-KR 인코딩 기준 60바이트 까지 허용 됩니다. contents는 EUC-KR 인코딩 기준 최대 2000자까지 전송 가능합니다. 2000자가 넘는 긴 문자열을 보낼 경우 2000자로 자동으로 잘려서 발송 됩니다.

주석: EUC-KR기준 80바이트를 넘지 않더라도 LMS URI로 Request 할 경우 장문 전송으로 과금 처리 됩니다.

주석: 기타 고려사항 및 Response는 *SMS발송결과* 와 동일합니다.

2.6 메시지 발송 결과 조회

HTTP Request Header:

HTTP Header	Value
Method	Get
URI Path	/smscenter/v1.0/sendresult/발송ID

주석: SMS또는 LMS 발송 결과로 리턴받은 발송ID를 URI에 붙여서 Request 하십 시오.

주석: 발송ID는 일정 기간 후에는 자동 소멸 되며 이후 발송 결과 조회에 사용할 수 없습니다.

Example:

```
Get /smscenter/v1.0/sendresult/10001 HTTP/1.1
```

HTTP Response Header:

HTTP Header	Value
Status	200
Content-Type	application/json; charset=utf-8

Response Data Format (JSON):

Key	Value	Format	비고
sent_time	“2014-06-12T16:01:45.923Z”	UTC Date	발송 시간
destination	“01011112222”	String	수신 번호
status	0	Int	*발송상태*

Example:

```
{status: 0, sent_time: "2014-06-12T16:01:45.923Z", destination: "01011112222"}
```

주석:

발송상태 표

status	의미
-1	확인 불가 (재시도 필요)
0	성공
10000	실패 (알수 없는 이유)
10001	서비스 불가 단말기
10002	NPDB (번호이동DB) 관련 에러
10003	서비스 일시 정지
10004	단말기 문제
10005	System 에러
10006	발송 제한시간 초과

2.7 메시지 발송 결과 조회 (그룹)

HTTP Request Header:

HTTP Header	Value
Method	Get
URI Path	/smscenter/v1.0/sendresult/group/그룹ID

주석: SMS또는 LMS 발송 결과로 리턴받은 그룹ID를 URI에 붙여서 Request 하십 시오.

Example:

```
Get /smscenter/v1.0/sendresult/group/10001 HTTP/1.1
```

HTTP Response Header:

HTTP Header	Value
Status	200
Content-Type	application/json; charset=utf-8

Response Data Format (JSON):

Key	Value	Format	비고
succeed	[”10001”, “10002”,]	List	필수
failed	[”10003”, “10004”,]	List	옵션
pending	[”10005”, “10006”,]	List	옵션

succeed, failed, pending 은 각각 성공된 발송ID 리스트, 실패한 발송ID리스트, 아직 성공여부가 확인 되지 않은 발송ID리스트 입니다. pending의 경우, 수신자가 응영지역에 있거나, 전원이 꺼져있는 등의 이유로 48시간 이내에 전달이 안되었 거나 아직 시스템에 의해 발송 성공 여부의 파악이 완료되지 않은 경우 입니다.

주석: **‘발송 결과 Callback‘_** 이 호출되기 이전에는 아직 시스템에 해당 그룹의 메시지의 발송 결과값들이 모두 파악 되지 않은 상태이기 때문에 대부분 pending 상태의 결과 값을 받게 됩니다.

2.8 예약 발송 취소

HTTP Request Header:

HTTP Header	Value
Method	Delete
URI Path	/smscenter/v1.0/cancel/예약ID

주석: SMS또는 LMS 발송 결과로 리턴받은 발송ID를 URI에 붙여서 Request 하십 시오.

Example:

```
Delete /smscenter/v1.0/cancel/187c4b7affa011e3bf47c42c032b413b
```

주석: 이 Request는 Response Data (Content) 없이 HTTP Status 로 결과가 리턴됩니다. HTTP Status Code 별 의미는 아래 표와 같습니다.

Status	의미
204	성공 (200과 같으나 content가 없으므로 HTTP 규약에 따라 204)
404	발송ID 찾을 수 없음
500	알수 없는 에러 (Internal Server Error)

2.9 수신거부 번호 조회

080 수신자 부담 전화 서비스를 통해 접수된 SMS 수신 거부 등록된 번호 목록을 조회하는 기능입니다.

HTTP Request Header:

HTTP Header	Value
Method	Get
URI Path	/smscenter/v1.0/rejectednumbers

Example:

```
Get /smscenter/v1.0/rejectednumbers HTTP/1.1
```

HTTP Response Header:

HTTP Header	Value
Status	200
Content-Type	application/json; charset=utf-8

Response Data Format (JSON):

Key	Value	Format
rejected	[수신거부Object, 수신거부Object,]	List

주석: 수신거부 Object의 Format은 다음과 같습니다.

Key	Value	Format	비고
number	“0101112222”	String	전화번호
registered	“2014-06-12T16:01:45.923Z”	UTC Date	수신거부 등록시간

Example:

```
{rejected: [
      {number: "01011112222", registered: "2014-06-12T16:01:45.923Z"},
      {number: "01011112223", registered: "2014-06-12T16:01:45.923Z"},
      {number: "01011112224", registered: "2014-06-12T16:01:45.923Z"},
]}
```

CHAPTER 3

Callback API

Callback API는 청기와랩의 Callback Gateway를 통해 고객의 웹서버로 실시간 정보 전달하기 위해 정의된 Open API입니다.

3.1 발송 결과 Callback

callback 값을 URL로 입력하면 전송 결과를 일정 시간 뒤 해당 URL로 보내줍니다. (발송 량과 상황에 따라 callback URL이 호출되는 시간은 발송 후 30초에서 최대 48시간 까지 소요될 수 있음)

Callback Request Header:

HTTP Header	Value
Method	Post
URI Path	your callback URI
Content-Type	application/json; charset=utf-8

Callback Request Data Format (JSON):

Key	Value	Format	비고
succeed	[”10001”, “10002”,]	List	필수
failed	[”10003”, “10004”,]	List	옵션
pending	[”10005”, “10006”,]	List	옵션

succeed, failed, pending 은 각각 성공된 발송ID 리스트, 실패한 발송ID리스트, 아직 성공여부가 확인 되지 않은 발송ID리스트 입니다. pending의 경우, 수신자가 응영지역에 있거나, 전원이 꺼져있는 등의 이유로 48시간 이내에 전달이 안된 경우 입니다. 이 경우 48시간 이내에 수신자가 수신가능 상태가 될경우 전송이 재개되어 추후 succeed 로 처리 되어 48시간 후 다시 callback URL로 succeed 상태로 report 될 수 있습니다. 또는 48시간이 경과 했음에도 전송이 되지 못했다면 48시간 후에 다시 callback URL로 failed 상태로 report 될 수 있습니다.

3.2 080 수신거부 Callback

한국의 정보통신망법 스팸 규제 법률에 따라 영리 목적으로 발송하는 매체에 080 수신자부담 전화 서비스를 통한 수신거부처리 서비스를 반드시 제공해야 합니다. 이를 위해 청기와랩은 수신거부 Callback 서비스를 제공합니다. 청기와랩의 080 수신거부 서비스를 통해 접수된 전화번호를 등록한 Callback URL로 실시간 전달 받으실 수 있습니다.

Callback Request Header:

HTTP Header	Value
Method	Post
URI Path	your callback URI for 080 Rejection
Content-Type	application/json; charset=utf-8

Callback Request Data Format (JSON):

Key	Value	Format	비고
rejected	[”0101112222”,]	List	필수

CHAPTER 4

Sample Codes

청기와랩의 SMS OpenAPI 샘플 코드들은 GitHub (https://github.com/bluehouselab/sms-openapi) 에서 다운 받으실 수 있습니다.

Git를 이용한 샘플코드 프로젝트 체크아웃:

```
$ git clone https://github.com/BlueHouseLab/sms-openapi.git
```

또는 샘플코드 Zip 아카이브 파일 다운 받기:

```
$ wget https://github.com/BlueHouseLab/sms-openapi/archive/master.zip
```

4.1 cURL

curl 은 많은 Unix 기반 Platform에 기본으로 탑재된 command line http client 입니다. 각종 Linux 배포판, BSD 배포판 및 Mac OS X에서 테스트 용도로 활용 가능 합니다.

env.sh - 발급 받은 appid, apikey 및 수발신 번호를 설정해 주세요.

```
#!/bin/bash
APPID=example
APIKEY=c5dd7e7dkjp273771903c42c032b413b
HOST=https://api.bluehouselab.com:443
```

sendsms.sh - sms 발송

```
#!/bin/bash

SENDER=01000000000 # FIXME MUST BE CHANGED AS REAL PHONE NUMBER
RECEIVER=01000000000 # FIXME MUST BE CHANGED AS REAL PHONE NUMBER

. env.sh # load env

curl -i -u $APPID:$APIKEY \
```

```
    -X POST $HOST/smscenter/v1.0/sendsms \
    -H "Content-Type: application/json;charset=utf-8" \
    -d '{"sender": "'$SENDER'", "receivers": ["'$RECEIVER'"], "content": "나는 유리를 먹
을 수 있어요. 그래도 아프지 않아요"}'
```

실행 예

```
$ ./sendsms.sh
HTTP/1.0 200 OK
Date: Thu, 03 Jul 2014 17:15:16 GMT
Content-Type: application/json; charset=utf-8

{"filtered": [], "reserved": null, "sent": [["01000000000", "20140704021516368253O"]]}
```

result.sh - 발송 확인 예제, sendsms.sh 의 결과로 받은 발송ID를 인자로 주어야 함

```
#!/bin/bash
. env.sh # load env

if [ $# != 1 ]; then
        echo Usage $0 deliveryid
        exit 1
fi

DELIVERYID=$1

curl -i -u $APPID:$APIKEY $HOST/smscenter/v1.0/result/$DELIVERYID
```

실행 예

```
$ ./result.sh 20140704021516368253O
HTTP/1.0 200 OK
Date: Thu, 03 Jul 2014 17:20:55 GMT
Content-Type: application/json; charset=utf-8

{"status": "0", "destination": "01000000000", "sent_time": "2014-07-03T17:15:20Z"}
```

4.2 Python

Python 예제는 Standard Library인 httplib 모듈을 이용하여 작성 되었습니다. httplib 모듈은 SSL Certificate Verification 과정이 생략된 HTTP Client 모듈 이므로 개발 참고용으로만 사용하세요. 실제 개발에는 보안을 위해 Requests 모듈을 추천 합니다.

conf.py - 발급 받은 appid, apikey 및 수발신 번호를 설정해 주세요.

```
# -*- coding: utf-8 -*-
import base64

appid = 'example'
apikey = 'c5dd7e7dkjp273771903c42c032b413b'
address = 'api.bluehouselab.com'

sender = '01000000000'          # FIXME - MUST BE CHANGED AS REAL PHONE NUMBER
receivers = ['01000000000', ] # FIXME - MUST BE CHANGED AS REAL PHONE NUMBERS
content = u'나는 유리를 먹을 수 있어요. 그래도 아프지 않아요'
```

```
credential = "Basic "+base64.encodestring(appid+':'+apikey).strip()
headers = {
  "Content-type": "application/json;charset=utf-8",
  "Authorization": credential,
}
```

sendsms.py - sms 발송

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import httplib
import json
from conf import address, sender, receivers, headers, content

c = httplib.HTTPSConnection(address)

path = "/smscenter/v1.0/sendsms"
value = {
    'sender'     : sender,
    'receivers'  : receivers,
    'content'    : content,
}
data = json.dumps(value, ensure_ascii=False).encode('utf-8')

c.request("POST", path, data, headers)
r = c.getresponse()

print r.status, r.reason
print r.read()
```

실행 예

```
$ python sendsms.py
200 OK
{"filtered": [], "reserved": null, "sent": [["01000000000", "201407040043284939320"]]}
```

result.py - 발송 확인 예제, sendsms.py 의 결과로 받은 발송ID를 인자로 주어야 함

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import httplib
import json
import sys

from conf import address, sender, receivers, headers, content

c = httplib.HTTPSConnection(address)

if len(sys.argv) == 1:
    print 'Usage:\n python %s deliveryid' % sys.argv[0]
    sys.exit(1)

deliveryid = sys.argv[1]
path = "/smscenter/v1.0/result/"+deliveryid

c.request("GET", path, '', headers)
r = c.getresponse()
```

```
print r.status, r.reason
print r.read()
```

실행 예

```
$ python result.py 201407040043284939320
200 OK
{"status": "0", "destination": "01000000000", "sent_time": "2014-07-03T15:43:31Z"}
```

4.3 Python (Requests)

Requests 모듈을 이용한 더 안전하고 Human을 위한 예제 구현 입니다.

sendsms.py - sms 발송

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import requests
import json
from conf import appid, apikey, sender, receivers, content

url = 'https://api.bluehouselab.com/smscenter/v1.0/sendsms'
params = {
    'sender'     : sender,
    'receivers'  : receivers,
    'content'    : content,
}
headers = {'Content-type': 'application/json; charset=utf-8',}
r = requests.post(url, data=json.dumps(params),
                auth=(appid, apikey), headers=headers)

print r.status_code, r.reason
if r.status_code == 200:
    print r.json()
```

실행 예

```
$ python sendsms.py
200 OK
{"filtered": [], "reserved": null, "sent": [["01000000000", "201407040043284939320"]]}
```

result.py - 발송 확인 예제, sendsms.py 의 결과로 받은 발송ID를 인자로 주어야 함

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import json
import requests
import sys
from conf import appid, apikey, sender, receivers, content

if len(sys.argv) == 1:
    print 'Usage:\n python %s deliveryid' % sys.argv[0]
    sys.exit(1)

deliveryid = sys.argv[1]
```

```
url = 'https://api.bluehouselab.com/smscenter/v1.0/result/'+deliveryid
params = {
    'sender'     : sender,
    'receivers'  : receivers,
    'content'    : content,
}
headers = {'Content-type': 'application/json; charset=utf-8',}
r = requests.get(url, auth=(appid, apikey), headers=headers)

print r.status_code, r.reason
if r.status_code == 200:
    print r.json()
```

실행 예

```
$ python result.py 201407040043284939320
200 OK
{"status": "0", "destination": "01000000000", "sent_time": "2014-07-03T15:43:31Z"}
```

4.4 Ruby

sendsms.rb - sms 발송

```
#!/usr/bin/env ruby

require 'net/https'
require 'uri'
require 'json'
require './conf.rb'

uri = URI.parse('https://api.bluehouselab.com/smscenter/v1.0/sendsms')

http = Net::HTTP.new(uri.host, uri.port)
http.use_ssl = true
req = Net::HTTP::Post.new(uri.request_uri, initheader={'Content-Type' =>'application/
↪json; charset=utf-8'})

req.basic_auth Conf::APPID, Conf::APIKEY
req.body = {
  sender: Conf::SENDER,
  receivers: Conf::RECEIVERS,
  content: Conf::CONTENT
}.to_json


resp = http.request(req)
puts resp.code
puts resp.body
```

실행 예

```
$ ruby sendsms.rb
200
{"filtered": [], "reserved": null, "sent": [["01000000000", "201407201409478863550
↪"]]]}
```

result.rb - 발송 확인 예제, sendsms.rb 의 결과로 받은 발송ID를 인자로 주어야 함

```
#!/usr/bin/env ruby

require 'net/https'
require 'uri'
require './conf.rb'
if ARGV.size != 1
        puts 'Usage:'
        puts ' '+__FILE__+' delivery-id'
        exit
end
did = ARGV[0]

uri = URI.parse('https://api.bluehouselab.com/smscenter/v1.0/result/'+did)

http = Net::HTTP.new(uri.host, uri.port)
http.use_ssl = true
req = Net::HTTP::Get.new(uri.request_uri)
req.basic_auth Conf::APPID, Conf::APIKEY

resp = http.request(req)
puts resp.code
puts resp.body
```

실행 예

```
$ ./result.rb 201407201409478863550
200
{"status": "0", "destination": "01000000000", "sent_time": "2014-07-20T05:09:51Z"}
```

4.5 PHP

sendsms.php - sms 발송

```
<?php
include "conf.php";

$data = array(
                        "sender" => $SENDER,
                        "receivers" => $RECEIVERS,
                        "content" => $CONTENT,
                );

$ch = curl_init("https://api.bluehouselab.com/smscenter/v1.0/sendsms");
curl_setopt($ch, CURLOPT_TIMEOUT, 30);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, TRUE);
curl_setopt($ch, CURLOPT_USERPWD, "$APPID:$APIKEY");
curl_setopt($ch, CURLOPT_HTTPHEADER, Array("Content-Type: application/json;␣
↪charset=utf-8"));
curl_setopt($ch, CURLOPT_CUSTOMREQUEST, "POST");
curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($data));
```

```
$response = curl_exec($ch);
$response_header = curl_getinfo($ch);
echo $response_header["http_code"]."\n";
print_r(json_decode($response, true));
curl_close($ch);
?>
```

실행 예

```
$ php sendsms.php
200
Array
(
    [filtered] => Array
        (
        )
    [reserved] =>
    [sent] => Array
        (
            [0] => Array
                (
                    [0] => 01000000000
                    [1] => 201408010307369291320
                )
        )
)
```

result.php - 발송 확인 예제, sendsms.php 의 결과로 받은 발송ID를 인자로 주어야 함

```
<?php
include "conf.php";

if ($argc != 2) {
        error_log("Usage:\n  ".$argv[0]." delivery-id");
        exit(1);
}
$delivery_id = $argv[1];

$ch = curl_init("https://api.bluehouselab.com/smscenter/v1.0/result/".$delivery_id);
curl_setopt($ch, CURLOPT_TIMEOUT, 30);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, TRUE);
curl_setopt($ch, CURLOPT_USERPWD, "$APPID:$APIKEY");

$response = curl_exec($ch);
$response_header = curl_getinfo($ch);
echo $response_header["http_code"]."\n";
print_r(json_decode($response, true));
curl_close($ch);
?>
```

실행 예

```
$ php result.php 201408010307369291320
200
Array
(
    [status] => 0
```

```
    [destination] => 01000000000
    [sent_time] => 2014-07-31T18:07:38Z
)
```

4.6 Javascript (Node.js)

sendsms.js - sms 발송

```
var c = require('./conf');
var https = require("https");
var credential = 'Basic '+new Buffer(c.APPID+':'+c.APIKEY).toString('base64');


var data = {
  "sender"     : c.SENDER,
  "receivers"  : c.RECEIVERS,
  "content"    : c.CONTENT
}
var body = JSON.stringify(data);

var options = {
  host: 'api.bluehouselab.com',
  port: 443,
  path: '/smscenter/v1.0/sendsms',
  headers: {
    'Authorization': credential,
    'Content-Type': 'application/json; charset=utf-8',
    'Content-Length': Buffer.byteLength(body)
  },
  method: 'POST'
};
var req = https.request(options, function(res) {
  console.log(res.statusCode);
  var body = "";
  res.on('data', function(d) {
    body += d;
  });
  res.on('end', function(d) {
          if(res.statusCode==200)
                console.log(JSON.parse(body));
        else
                console.log(body);
  });
});
req.write(body);
req.end();
req.on('error', function(e) {
        console.error(e);
});
```

실행 예

```
$ node sendsms.js
200
{ filtered: [],
```

```
  reserved: null,
  sent: [ [ '01000000000', '201408021612156351310' ] ] }
```

result.js - 발송 확인 예제, sendsms.js 의 결과로 받은 발송ID를 인자로 주어야 함

```
var c = require('./conf');
var https = require("https");
var credential = 'Basic '+new Buffer(c.APPID+':'+c.APIKEY).toString('base64');
if(process.argv.length != 3) {
        console.log('Usage: ');
        console.log(' node result.js delivery-id');
        process.exit(1);
}
var deliveryid = process.argv[2];

var options = {
  host: 'api.bluehouselab.com',
  port: 443,
  path: '/smscenter/v1.0/result/'+deliveryid,
  headers: {
    'Authorization': credential,
  },
  method: 'GET'
};
var req = https.request(options, function(res) {
  console.log(res.statusCode);
  var body = "";
  res.on('data', function(d) {
    body += d;
  });
  res.on('end', function(d) {
          if(res.statusCode==200)
                  console.log(JSON.parse(body));
        else
                  console.log(body);
  });
});
req.end();
req.on('error', function(e) {
        console.error(e);
});
```

실행 예

```
$ node result.js 201408021612156351310
200
{ status: '10001',
  destination: '01000000000',
  sent_time: '2014-08-01T18:14:35Z' }
```

4.7 Java (Android)

Java 예제는 Apache 프로젝트의 Http Component 를 사용하여 작성 하였 습니다. Mac, Linux, Win32 등의 일반적인 Java 개발 환경 뿐만 아니라, Apache Http Component가 기본 탑재 되어 있는 Android (API level 1) 에도 적용 가능한 샘플 입니다.

빌드 (Linux/Mac/Win32) - GNU make 명령을 이용하여 Java 코드들을 컴파일 합니다. Makefile 을 참고 하세요.

```
$ make
```

Sendsms.java - sms 발송

```
import org.apache.http.client.protocol.HttpClientContext;



/* XXX didn't use org.json to be simple
import org.json.JSONObject;
*/

public final class SendSMS {
    public static void main(String[] args) {
        String hostname = "api.bluehouselab.com";
        String url = "https://"+hostname+"/smscenter/v1.0/sendsms";

        CredentialsProvider credsProvider = new BasicCredentialsProvider();
        credsProvider.setCredentials(
            new AuthScope(hostname, 443, AuthScope.ANY_REALM),
            new UsernamePasswordCredentials(Config.appid, Config.apikey)
        );

        // Create AuthCache instance
        AuthCache authCache = new BasicAuthCache();
        authCache.put(new HttpHost(hostname, 443, "https"), new BasicScheme());

        // Add AuthCache to the execution context
        HttpClientContext context = HttpClientContext.create();
        context.setCredentialsProvider(credsProvider);
        context.setAuthCache(authCache);

        DefaultHttpClient client = new DefaultHttpClient();

        try {
            HttpPost httpPost = new HttpPost(url);
            httpPost.setHeader("Content-type", "application/json; charset=utf-8");
            String json = "{\"sender\":\""+Config.sender+"\",\"receivers\":[\"
↪"+Config.receiver+"\"],\"content\":\""+Config.content+"\"}";

            StringEntity se = new StringEntity(json, "UTF-8");
            httpPost.setEntity(se);

            HttpResponse httpResponse = client.execute(httpPost, context);
            System.out.println(httpResponse.getStatusLine().getStatusCode());

            InputStream inputStream = httpResponse.getEntity().getContent();
            if(inputStream != null) {
                BufferedReader bufferedReader = new BufferedReader( new␣
↪InputStreamReader(inputStream));
                String line = "";
                while((line = bufferedReader.readLine()) != null)
                    System.out.println(line);
                inputStream.close();
            }
        } catch (Exception e) {
            System.err.println("Error: "+e.getLocalizedMessage());
```

```
        } finally {
            client.getConnectionManager().shutdown();
        }

    }
}
```

실행 예

```
$ ./sendsms # bash shell script to execute java with require params
200
{"filtered": [], "reserved": null, "sent": [["01000000000", "201408030412368846800"]]}
```

Result.java - 발송 확인 예제, SendSMS.java 의 결과로 받은 발송ID를 인자로 주어야 함

```
import org.apache.http.client.AuthCache;
import org.apache.http.HttpHost;
import org.apache.http.client.protocol.HttpClientContext;


public final class Result {
    public static void main(String[] args) {
        if(args.length != 1) {
            System.err.println("Usage:");
            System.err.println("  ./result delivery-id");
            return;
        }
        String hostname = "api.bluehouselab.com";
        String url = "https://"+hostname+"/smscenter/v1.0/result/"+args[0];

        CredentialsProvider credsProvider = new BasicCredentialsProvider();
        credsProvider.setCredentials(
            new AuthScope(hostname, 443, AuthScope.ANY_REALM),
            new UsernamePasswordCredentials(Config.appid, Config.apikey)
        );

        // Create AuthCache instance
        AuthCache authCache = new BasicAuthCache();
        authCache.put(new HttpHost(hostname, 443, "https"), new BasicScheme());

        // Add AuthCache to the execution context
        HttpClientContext context = HttpClientContext.create();
        context.setCredentialsProvider(credsProvider);
        context.setAuthCache(authCache);

        DefaultHttpClient client = new DefaultHttpClient();

        try {
            HttpGet httpGet = new HttpGet(url);

            HttpResponse httpResponse = client.execute(httpGet, context);
            System.out.println(httpResponse.getStatusLine().getStatusCode());

            InputStream inputStream = httpResponse.getEntity().getContent();
            if(inputStream != null) {
                BufferedReader bufferedReader = new BufferedReader( new␣
↪InputStreamReader(inputStream));
                String line = "";
```

```
                while((line = bufferedReader.readLine()) != null)
                    System.out.println(line);
                inputStream.close();
            }
        } catch (Exception e) {
            System.err.println("Error: "+e.getLocalizedMessage());
        } finally {
            client.getConnectionManager().shutdown();
        }

    }
}
```

실행 예

```
$ ./result 201408030412368846800 # bash shell script to execute java
200
{"status": 0, "destination": "01000000000", "sent_time": "2014-08-02T19:12:35Z"}
```

4.8 C (libcurl)

C 예제는 libcurl 라이브러리를 HTTP Client 로 사용하였습니다. libcurl은 Linux/BSD/Win32 환경에서 가장 많이 사용되는 통신 모듈 중 하나로 이식성이 우수 하며 여러 프로젝트에서 사용되어 안정성 및 그 우수성을 인정 받고 있는 오픈 소스 라이브러리 입니다.

빌드 (Linux/Mac/Win32) - GNU make 및 pkg-config 명령을 이용하여 빌드 합니다. Win32 환경에서는 cygwin 또는 msys 환경에서 쉽게 빌드 하실 수 있습니다.

```
$ make
cc -Wall -O2 -o sendsms.o -c sendsms.c `pkg-config --cflags libcurl`
cc -o sendsms sendsms.o `pkg-config --libs libcurl`
cc -Wall -O2 -o result.o -c result.c `pkg-config --cflags libcurl`
cc -o result result.o `pkg-config --libs libcurl`
```

sendsms.c - sms 발송

```
int main(void)
{
  CURL *handle;
  CURLcode res;
  char post_body[BUFSIZ] = {0, };
  char responsebuf[BUFSIZ] = {0, };

  curl_global_init(CURL_GLOBAL_DEFAULT);

  handle = curl_easy_init();
  if(handle) {
    curl_easy_setopt(handle, CURLOPT_URL, "https://api.bluehouselab.com/smscenter/v1.
↪0/sendsms");
    curl_easy_setopt(handle, CURLOPT_USERNAME, APPID);
    curl_easy_setopt(handle, CURLOPT_PASSWORD, APIKEY);

    curl_easy_setopt(handle, CURLOPT_WRITEFUNCTION, write_callback);
    curl_easy_setopt(handle, CURLOPT_WRITEDATA, responsebuf);
```

```
    sprintf(post_body, "{\"sender\":\"%s\",\"receivers\":[\"%s\"],\"content\":\"%s\"}
↪", SENDER, RECEIVER, CONTENT);
    curl_easy_setopt(handle, CURLOPT_POST, 1);
    curl_easy_setopt(handle, CURLOPT_POSTFIELDS, post_body);
    curl_easy_setopt(handle, CURLOPT_POSTFIELDSIZE, strlen(post_body));

    res = curl_easy_perform(handle);
    if(res != CURLE_OK)
      fprintf(stderr, "curl_easy_perform() failed: %s\n",
              curl_easy_strerror(res));
    else {
        long status_code = 0;
        curl_easy_getinfo (handle, CURLINFO_RESPONSE_CODE, &status_code);
        printf("%ld\n", status_code);
        printf("%s\n", responsebuf);

    }

    curl_easy_cleanup(handle);
  }

  curl_global_cleanup();

  return 0;
}
```

실행 예

```
$ ./sendsms
200
{"filtered": [], "reserved": null, "sent": [["01000000000", "201408030412368846800"]]}
```

result.c - 발송 확인 예제, sendsms.c 의 결과로 받은 발송ID를 인자로 주어야 함

```
int main(int argc, char** argv)
{
  CURL *handle;
  CURLcode res;
  char responsebuf[BUFSIZ] = {0, };
  char url[BUFSIZ] = {0, };
  char *deliveryid = NULL;

  if(argc != 2) {
      fprintf(stderr, "Usage: \n");
      fprintf(stderr, "  %s delivery-id\n", argv[0]);
      return 1;
  }
  deliveryid = argv[1];

  curl_global_init(CURL_GLOBAL_DEFAULT);

  handle = curl_easy_init();
  if(handle) {
    sprintf(url,
            "https://api.bluehouselab.com/smscenter/v1.0/result/%s",
            deliveryid);
```

```
    curl_easy_setopt(handle, CURLOPT_URL, url);
    curl_easy_setopt(handle, CURLOPT_USERNAME, APPID);
    curl_easy_setopt(handle, CURLOPT_PASSWORD, APIKEY);

    curl_easy_setopt(handle, CURLOPT_WRITEFUNCTION, write_callback);
    curl_easy_setopt(handle, CURLOPT_WRITEDATA, responsebuf);


    res = curl_easy_perform(handle);
    if(res != CURLE_OK)
      fprintf(stderr, "curl_easy_perform() failed: %s\n",
              curl_easy_strerror(res));
    else {
        long status_code = 0;
        curl_easy_getinfo (handle, CURLINFO_RESPONSE_CODE, &status_code);
        printf("%ld\n", status_code);
        printf("%s\n", responsebuf);

    }

    curl_easy_cleanup(handle);
  }

  curl_global_cleanup();

  return 0;
}
```

실행 예

```
$ ./result 201408030412368846800
200
{"status": 0, "destination": "01000000000", "sent_time": "2014-08-02T19:12:35Z"}
```

4.9 C++ (Qt5)

C++ 예제는 Qt5 를 HTTP Client 로 사용하였습니다. Qt 는 오랜 기간 동안 오픈소스 소프트웨어 발전에 기여한 훌륭한 GUI 프레임워크 입니다. LGPL라이센스로 Linux(X11/Wayland), Mac(X11/Cocoa), Win32, iOS, Android, SailFish OS 환경에서 모두 사용 가능합니다. 이 예제는 GUI를 사용하지 않고 Console 환경에서 SMS를 발송 할수 있도록 심플하게 작성 되었습니다.

빌드 (Linux/Mac/Win32) - GNU make 및 Qt5의 qmake 명령을 이용하여 빌드 합니다.

```
$ make
```

sendsms.cpp - sms 발송

```
void SMSClient::sendSMS()
{
    QUrl url("https://api.bluehouselab.com/smscenter/v1.0/sendsms");
    QNetworkRequest request(url);
    request.setHeader(QNetworkRequest::ContentTypeHeader,
                      QVariant("application/json; charset=utf-8"));

    QJsonDocument json;
```

```
    QJsonObject obj;
    obj["sender"] = QString(SENDER);
    QJsonArray receivers;
    receivers.push_back(QString(RECEIVER));
    obj["receivers"] = receivers;
    obj["content"] = QString(CONTENT);
    json.setObject(obj);
    manager.post(request, json.toJson());
}
```

실행 예

```
$ ./sendsms
"{"filtered": [], "reserved": null, "sent": [["01000000000", "201408032346305478310
↪"]]}"
```

result.cpp - 발송 확인 예제, sendsms.cpp 의 결과로 받은 발송ID를 인자로 주어야 함

```
void SMSClient::checkResult()
{
    QStringList args = QCoreApplication::instance()->arguments();
    QString program = args.takeFirst();
    if (args.isEmpty()) {
        qCritical() << "Usage";
        qCritical() << QString("  %1 delivery-id").arg(program);
        QCoreApplication::instance()->quit();
        return;
    }
    QString deliveryid = args.takeFirst();
    QUrl url(QString("https://api.bluehouselab.com/smscenter/v1.0/result/%1").
↪arg(deliveryid));
    QNetworkRequest request(url);
    manager.get(request);
}
```

실행 예

```
$ ./result 201408032346305478310
200
"{"status": 0, "destination": "01000000000", "sent_time": "2014-08-03T14:46:29Z"}"
```

4.10 C# (Mono/.NET)

C# 예제는 System.Net.WebClient 를 사용하여 간단하게 작성 되었습니다. 또한 오픈소스 .NET 프레임워크인 Mono 환경에서 작성 및 테스트 되었습니다. Mono를 통해 Windows는 물론 Linux, Mac OS X 환경에서도 테스트 가능합니다.

빌드 (Linux/Mac) - GNU make를 이용하여 빌드 합니다.

```
$ make
```

sendsms.cs - sms 발송

```
class SendSMS
{
```

```
    static void Main(string[] args)
    {
        string url = "https://api.bluehouselab.com/smscenter/v1.0/sendsms";
        string appid = "example";
        string apikey = "c5dd7e7dkjp273771903c42c032b413b";

        string json = @"{
            ""sender"": ""01000000000"",
            ""receivers"": [""01000000000""],
            ""content"": ""나는 유리를 먹을 수 있어요. 그래도 아프지 않아요""}";

        WebClient client = new WebClient();
        NetworkCredential creds = new NetworkCredential(appid, apikey);
        client.Credentials = creds;
        client.Headers[HttpRequestHeader.ContentType] = "application/json;␣
↪charset=utf-8";

        try
        {
            string response = client.UploadString(url, json);
            Console.WriteLine(response);
        }
        catch (WebException e)
        {
            HttpStatusCode status = ((HttpWebResponse)e.Response).StatusCode;
            Console.WriteLine("{0}", (int)status);
            Console.WriteLine("{0}", status.ToString());
        }
    }
}
```

실행 예 (Mac OS X)

```
$ mono sendsms.exe
{"filtered": [], "reserved": null, "sent": [["01000000000", "201409130128257158920"]]}
```

result.cs - 발송 확인 예제, sendsms.cs 의 결과로 받은 발송ID를 인자로 주어야 함

```
class Result
{
    static void Main(string[] args)
    {
        if(args.Length != 1)
        {
            Console.WriteLine("Usage:");
            Console.WriteLine("  result.exe deliveryid");
            return;
        }
        string url = "https://api.bluehouselab.com/smscenter/v1.0/result/"+args[0];
        string appid = "example";
        string apikey = "c5dd7e7dkjp273771903c42c032b413b";

        WebClient client = new WebClient();
        NetworkCredential creds = new NetworkCredential(appid, apikey);
        client.Credentials = creds;

        try
        {
```

```
            byte[] response = client.DownloadData(url);
            Console.WriteLine(Encoding.UTF8.GetString(response));
        }
        catch (WebException e)
        {
            HttpStatusCode status = ((HttpWebResponse)e.Response).StatusCode;
            Console.WriteLine("{0}", (int)status);
            Console.WriteLine("{0}", status.ToString());
        }
    }
}
```

실행 예 (Mac OS X)

```
$ mono result.exe 201409130128257158920
{"status": "10001", "destination": "01000000000", "sent_time": "2014-09-12t16:28:25z"}
```

CHAPTER 5

Indices and tables

- genindex
- modindex
- search

Python 모듈 목록

색인