

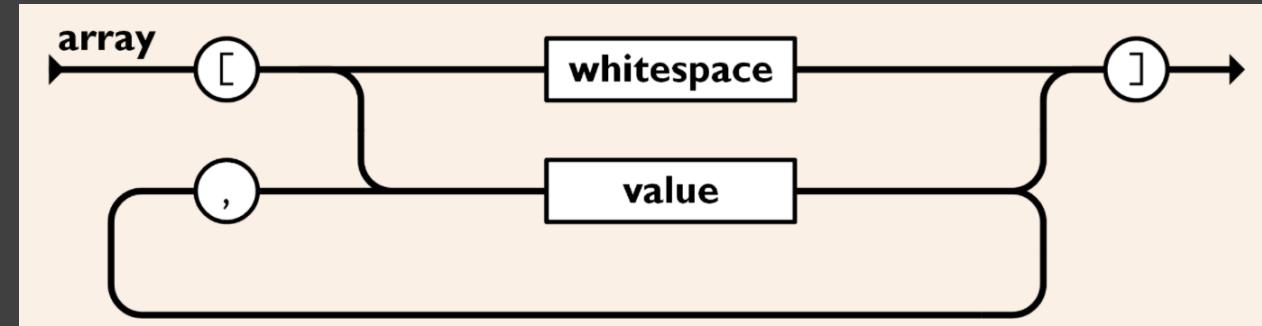
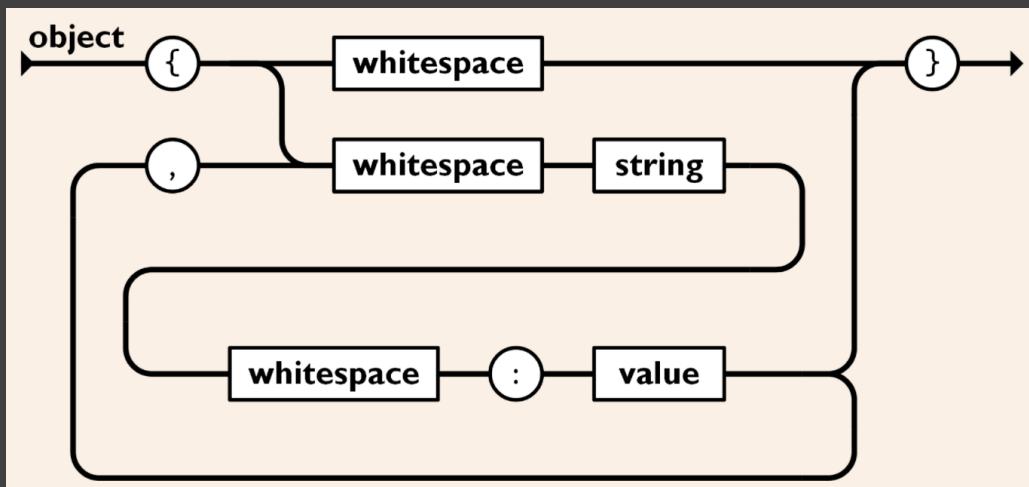
제13 강 Messaging

학습 목차

- JSON
- HTTP Request
- REST
- 텔리그램 메시지 주고받기
- 카카오톡 나에게 문자 보내기
- SMS 보내기

JSON(JavaScript Object Notation)

- JavaScript 프로그램의 표준 데이터 교환 형식
- 웹사이트들이 데이터를 JSON 형식으로 제공하는 경우가 많음.
- 파이썬 리스트 및 딕셔너리와 “거의” 동일
- 큰따옴표만 허용, true, false,



JSON 데이터 문자열 처리

```
import json

json_data_string = '[1,"Suji",true,false, 3.14]'
data_list = json.loads(json_data_string)

data_list.append('Rose')
json.dumps(data_list)
```

JSON 데이터 파일 처리

```
import json

suji_data = \
{
    "City" : "Seoul",
    "Name" : "Siji",
    "Age" : 22,
    "Height" : 165.5
}

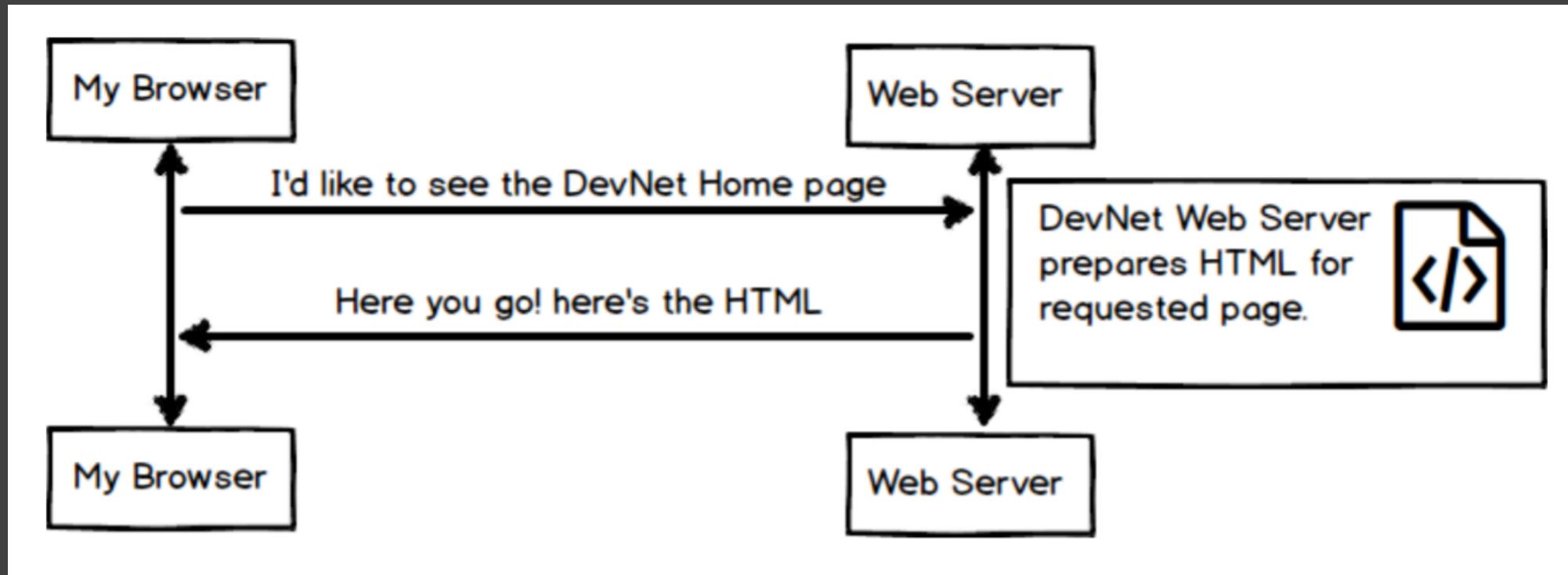
with open('sujidata.json', 'w') as f:
    json.dump(suji_data, f)

with open('sujidata.json') as f:
    data = json.load(f)

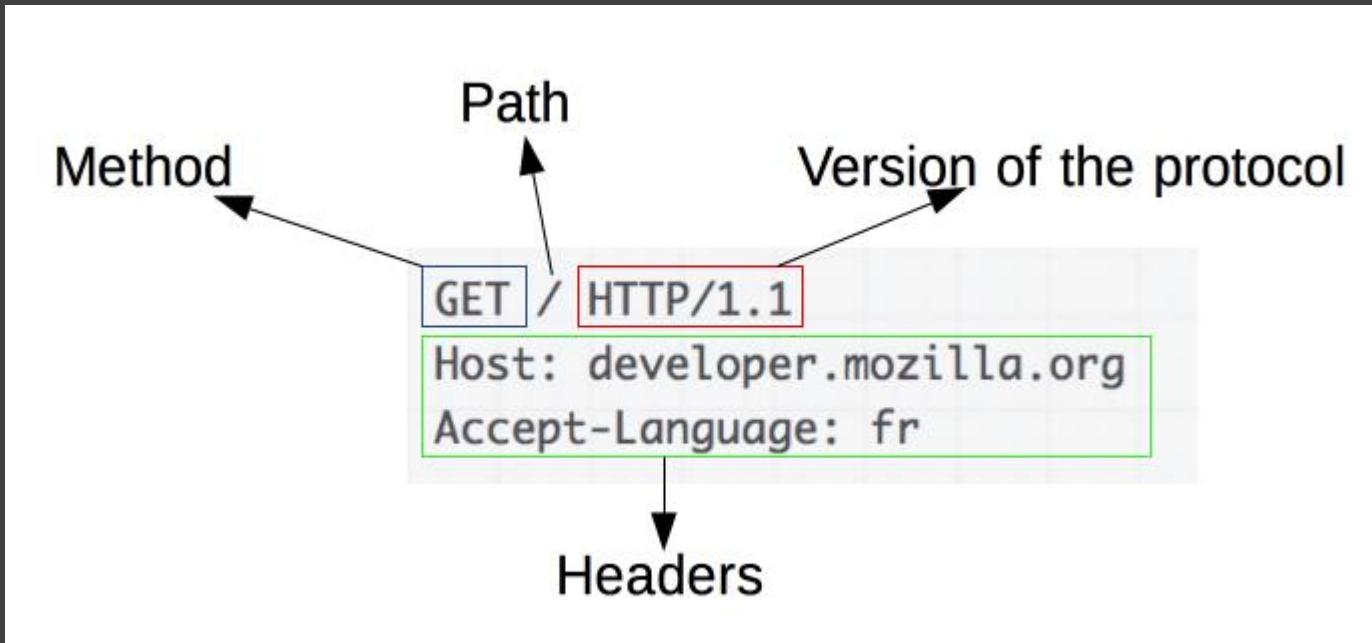
data
```

HTTP(HyperText Transfer Protocol)

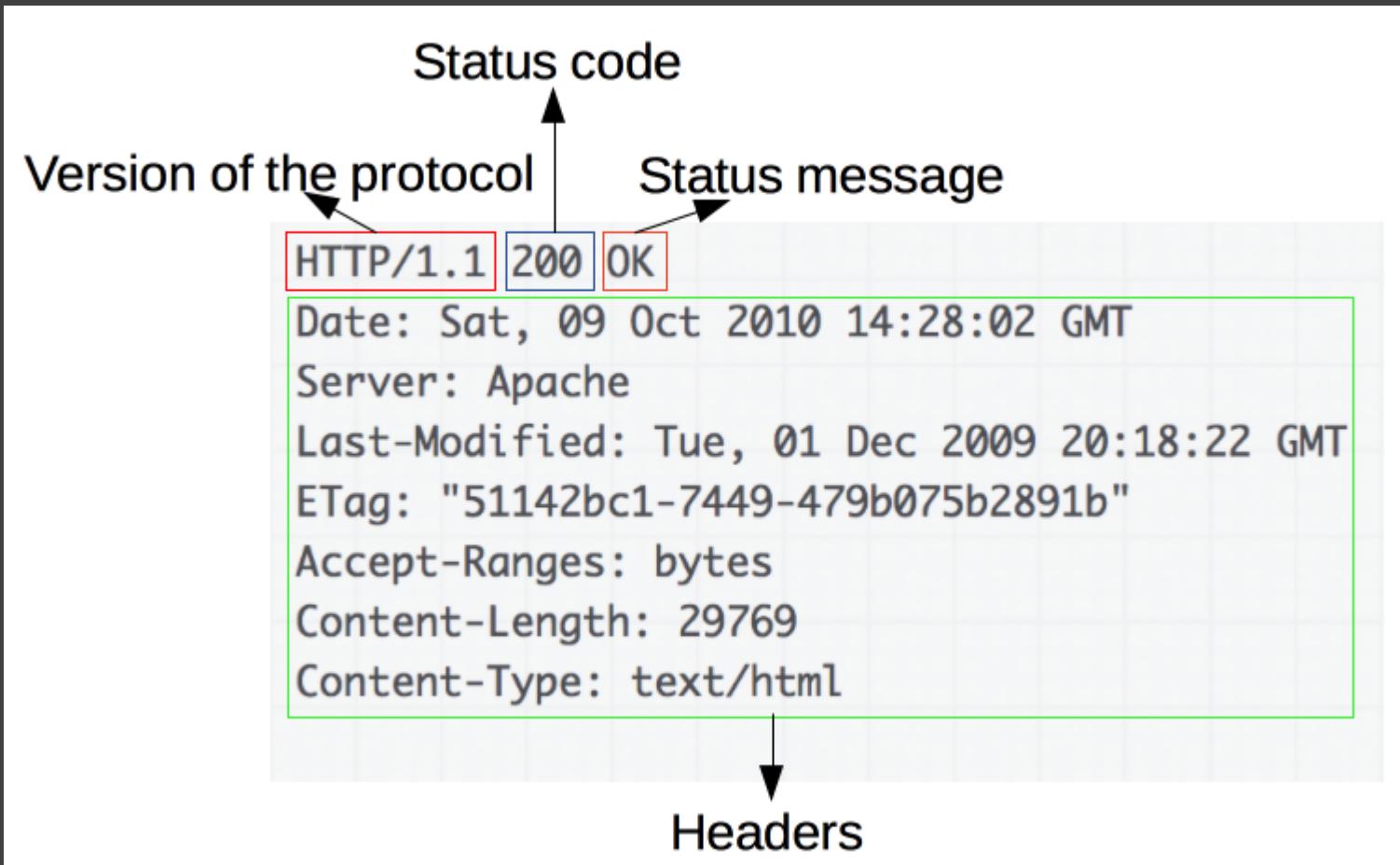
- HTML 문서를 주고 받는 통신 규약
- 요청(request)를 하면 응답(response)을 받음.



요청 - 서버에 있는 리소스에 대한 처리 요청



응답



요청(request)의 종류

- **GET**
 - 서버에 있는 리소스를 받아옴.
- **POST**
 - 서버에 리소스를 추가.
 - 동일한 요청은 리소스를 여러 개 만들어 냈.(생성)
- **PUT**
 - 서버에 리소스를 추가하거나 또는 업데이트(수정)
 - 동일한 요청에 대해서 결과가 동일.
- **DELETE**
 - 서버에 있는 리소스의 삭제

requests 모듈 – http library

```
import requests

def print_dict(d):
    for k, v in d.items():
        print(k, ':', v)

r = requests.get('https://httpbin.org')

r.status_code
r.headers
r.headers['content-type']

r.text
r.encoding
r.url
r.content #binary
```

GET : URL로 인자 전달

```
params = {'city':'Seoul', 'name':'Suji'}  
r = requests.get('https://httpbin.org/get', params=params)  
  
r.url  
# 'https://httpbin.org/get?city=Seoul&name=Suji'
```

URL 에 인자가 표시

POST

```
r = requests.post('https://httpbin.org/post', data={'key':'value'})
```

```
r.url
```

```
# 'https://httpbin.org/post'
```

URL 에 데이터 표시 X, 민감한 정보 제공.

JSON response

```
r = requests.post('https://httpbin.org/post', data = {'key':'value'})
```

```
r.headers
```

```
r.headers['Content-Type']
```

```
r.json()
```

API (Application Programming Interface)

- 소프트웨어 프로그램들이 서로 통신하는 방법
- 주요 API
 - REST
 - Representational State Transfer
 - HTTP 상에서 작동
 - XML, JSON 등의 형식이 이용됨.
 - SOAP
 - Simple Object Access Protocol
 - XML 형식 사용.
 - HTTP, SMTP, TCP, UDP 등을 이용하여 통신.
 - 컴퓨터 언어
 - Java, Python, C 등으로 직접 구현.

REST(Representational State Transfer)

- 구성
 - 리소스 : URI로 표시
 - 행위: HTTP method로 처리 – GET, POST, PUT, DELETE

METHOD	역할
POST	POST를 통해 해당 URI를 요청하면 리소스를 생성합니다.
GET	GET를 통해 해당 리소스를 조회합니다. 리소스를 조회하고 해당 도큐먼트에 대한 자세한 정보를 가져온다.
PUT	PUT를 통해 해당 리소스를 수정합니다.
DELETE	DELETE를 통해 리소스를 삭제합니다.

상태 코드

상태코드	
200	클라이언트의 요청을 정상적으로 수행함
201	클라이언트가 어떠한 리소스 생성을 요청, 해당 리소스가 성공적으로 생성됨(POST를 통한 리소스 생성 작업 시)
상태코드	
400	클라이언트의 요청이 부적절 할 경우 사용하는 응답 코드
401	클라이언트가 인증되지 않은 상태에서 보호된 리소스를 요청했을 때 사용하는 응답 코드 (로그인 하지 않은 유저가 로그인 했을 때, 요청 가능한 리소스를 요청했을 때)
403	유저 인증상태와 관계 없이 응답하고 싶지 않은 리소스를 클라이언트가 요청했을 때 사용하는 응답 코드 (403 보다는 400이나 404를 사용할 것을 권고. 403 자체가 리소스가 존재한다는 뜻이기 때문에)
405	클라이언트가 요청한 리소스에서는 사용 불가능한 Method를 이용했을 경우 사용하는 응답 코드
상태코드	
301	클라이언트가 요청한 리소스에 대한 URI가 변경 되었을 때 사용하는 응답 코드 (응답 시 Location header에 변경된 URI를 적어 줘야 합니다.)
500	서버에 문제가 있을 경우 사용하는 응답 코드

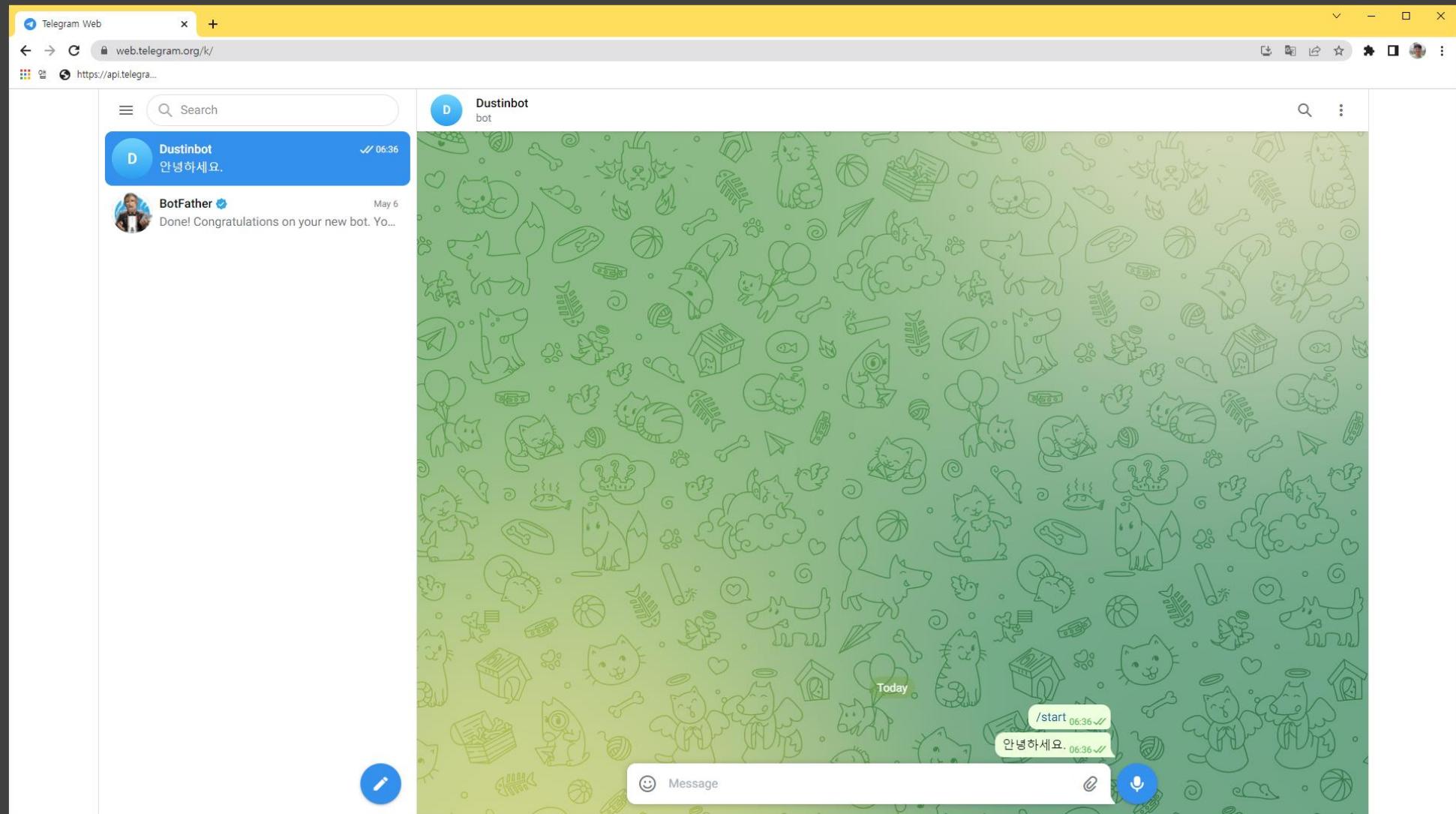
Telegram API

- HTTPS request 로 접근 가능
- Bot API
 - 봇을 이용해서, 사용자와 채팅 가능
 - 봇 : 전화번호 등이 필요없는 특별한 계정
 - 텔리그램 서버와 통신을 위해 복잡한 작업을 할 필요없음.
- Telegram API
 - 맞춤형 Telegram client 를 별도로 제작할 수 있음.

Telegra API Wrapper

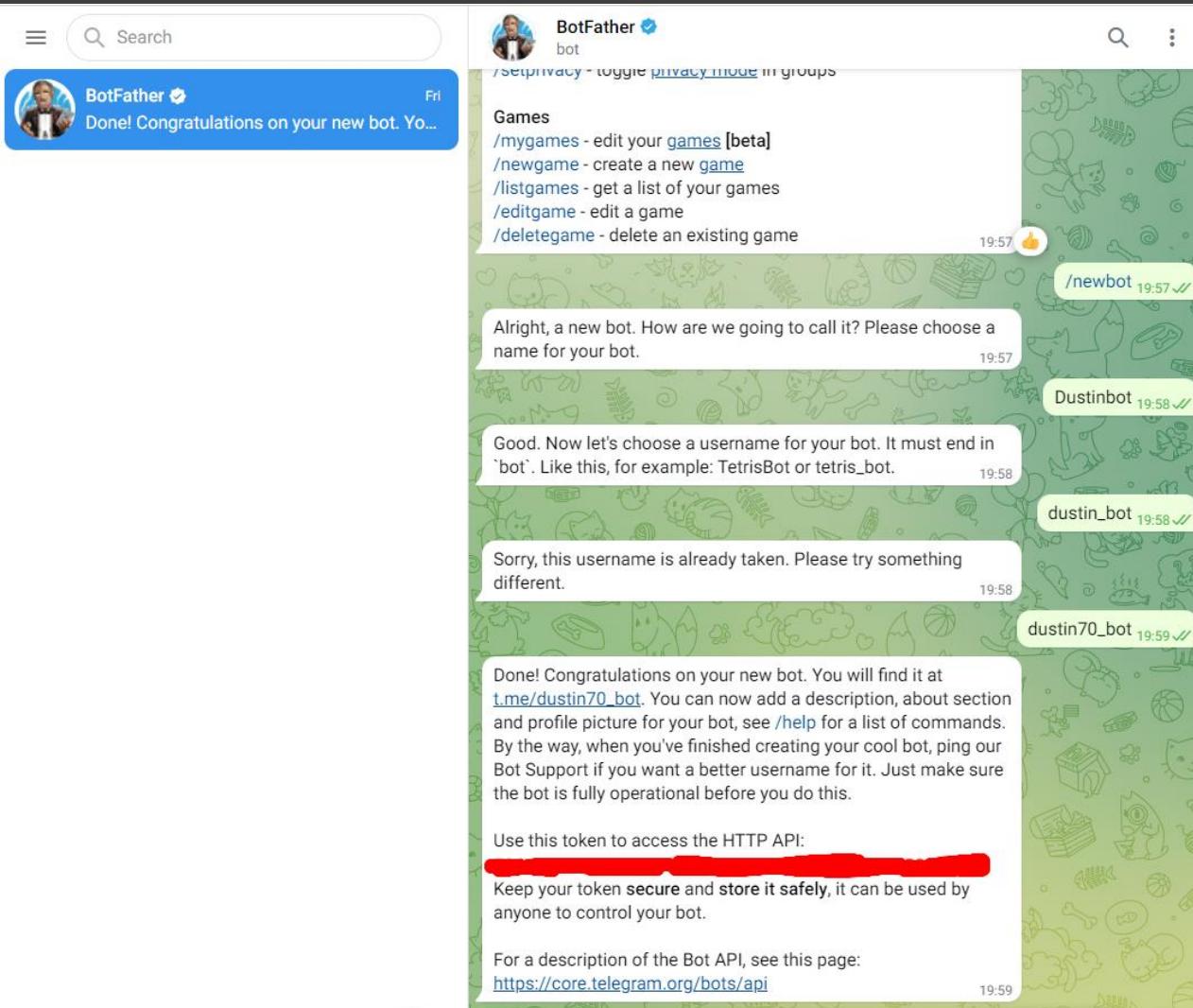
- **pyTelegramBotAPI**
- **python-telegram-bot**

Telegram Web - <https://web.telegram.org/k/>



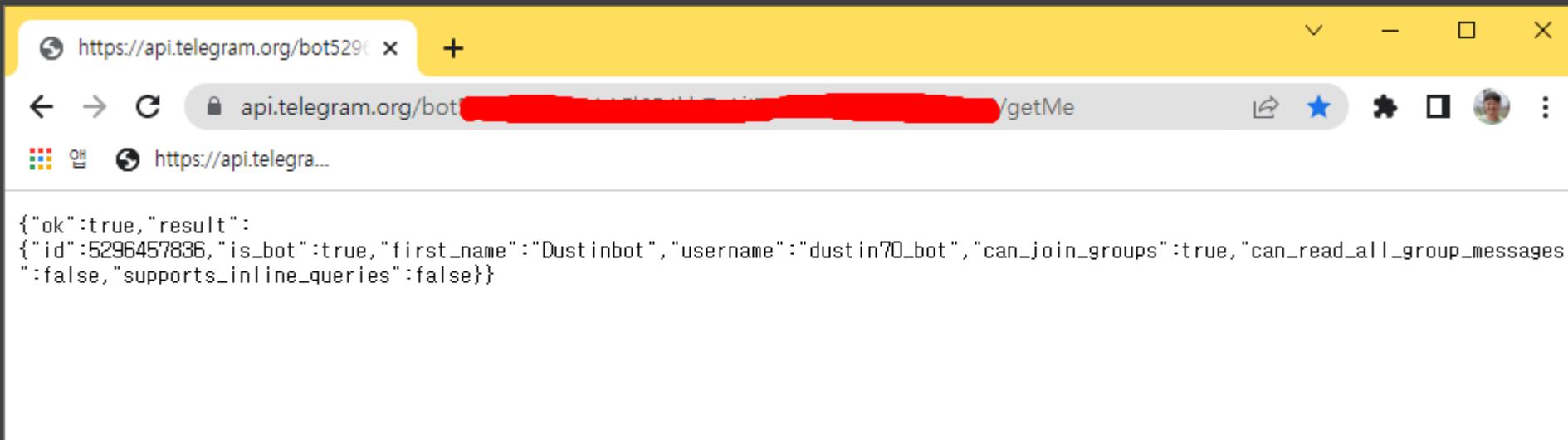
봇 계정 생성

- @BotFathter 와 대화를 통해서 봇 계정 생성



봇 계정 확인

- <https://api.telegram.org/botxxxxxx/getMe>



A screenshot of a web browser window. The address bar shows the URL `https://api.telegram.org/bot[REDACTED]/getMe`. The page content displays a JSON response:

```
{"ok":true,"result": {"id":5296457836,"is_bot":true,"first_name":"Dustinbot","username":"dustin70_bot","can_join_groups":true,"can_read_all_group_messages":false,"supports_inline_queries":false}}
```

```
import requests

# get information about me, or mybot
r = requests.get(f'https://api.telegram.org/bot{mybot.token}/getMe')
r.headers
r.headers['content-type']
r.json()
```

Telegram Bot API

The screenshot shows a web browser displaying the official Telegram Bot API documentation at core.telegram.org/bots/api. The page has a yellow header bar with the title "Telegram Bot API". Below the header, there's a navigation bar with links for "Home", "FAQ", "App", "API", "Protocols", and "Skim". On the right side of the header, there are social sharing icons for Twitter and other platforms.

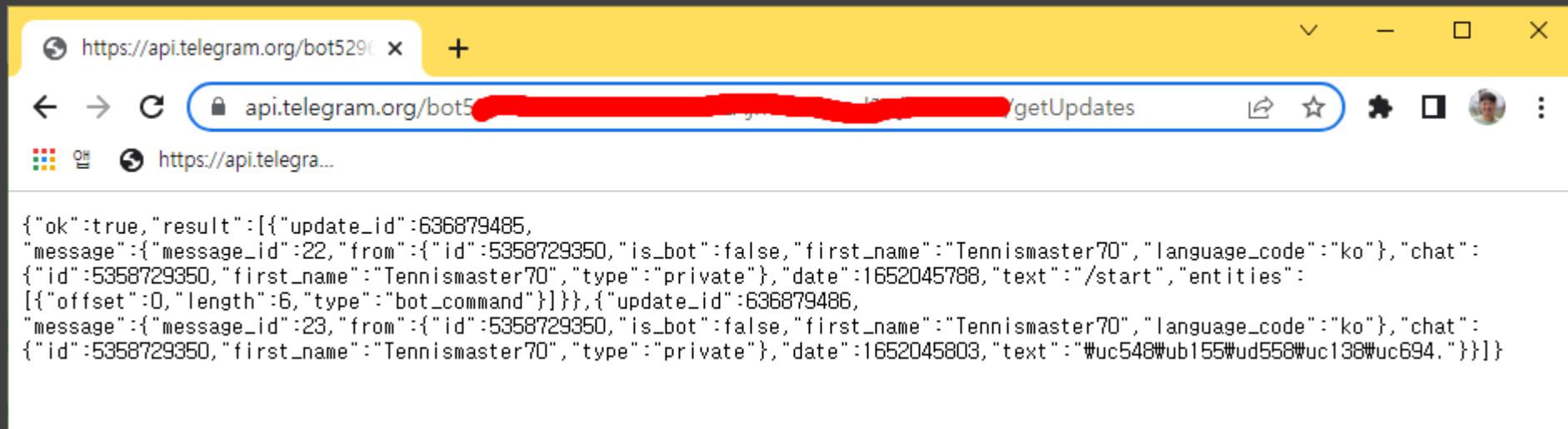
The main content area has a breadcrumb navigation bar at the top left: "Telegram Bots > Telegram Bot API". The main title is "Telegram Bot API". A sidebar on the right lists "Recent changes" with dates from April 21, 2023, to February 3, 2023, followed by a vertical list of API-related topics: "Authorizing your bot", "Making requests", "Using a Local Bot API S...", "Getting updates", "Available types", "Available methods", "Updating messages", "Stickers", "Inline mode", "Payments", "Telegram Passport", and "Games".

The main content includes a section titled "Recent changes" with a call to action to subscribe to @BotNews and join @BotTalk. It also lists recent changes dated April 21, 2023, which include:

- Added support for launching Web Apps from inline query results by replacing the parameters `switch_pm_text` and `switch_pm_parameter` of the method `answerInlineQuery` with the parameter `button` of type `InlineQueryResultsButton`.
- Added the field `web_app_name` to the class `WriteAccessAllowed`.
- Added the field `switch_inline_query_chosen_chat` of the type `SwitchInlineQueryChosenChat` to the class `InlineKeyboardButton`, which allows bots to switch to inline mode in a chosen chat of the given type.
- Added the field `via_chat_folder_invite_link` to the class `ChatMemberUpdated`.
- Added the ability to set different bot names for different user languages using the method `setMyName`.
- Added the ability to get the current bot name in the given language as the class `BotName` using the method `getMyName`.
- Added the ability to change bot settings from the bot's profile in official Telegram apps, including the ability to

메시지 확인 (봇이 받은)

- <https://api.telegram.org/botxxxxxx/getUpdates>



json 해석

```
{  
    "ok": true,  
    "result": [  
        {  
            "update_id": 296381954,  
            "message": {  
                "message_id": 1,  
                "from": {  
                    "id": 5358729350,  
                    "is_bot": false,  
                    "first_name": "\ub300\ud604",  
                    "last_name": "\uc774",  
                    "username": "Tennismaster70",  
                    "language_code": "ko"  
                },  
                "chat": {  
                    "id": 5358729350,  
                    "first_name": "\ub300\ud604",  
                    "last_name": "\uc774",  
                    "username": "Tennismaster70",  
                    "type": "private"  
                },  
                "date": 1682769798,  
                "text": "/start",  
                "entities": [  
                    {  
                        "offset": 0,  
                        "length": 6,  
                        "type": "bot_command"  
                    }  
                ]  
            },  
            "update_id": 296381955,  
            "message": {  
                "message_id": 2,  
                "from": {  
                    "id": 5358729350,  
                    "is_bot": false,  
                    "first_name": "\ub300\ud604",  
                    "last_name": "\uc774",  
                    "username": "Tennismaster70",  
                    "language_code": "ko"  
                },  
                "chat": {  
                    "id": 5358729350,  
                    "first_name": "\ub300\ud604",  
                    "last_name": "\uc774",  
                    "username": "Tennismaster70",  
                    "type": "private"  
                },  
                "date": 1682769798,  
                "text": "Hello! How can I help you today?",  
                "entities": [  
                    {  
                        "offset": 0,  
                        "length": 18,  
                        "type": "bot_command"  
                    }  
                ]  
            }  
        }  
    ]  
}
```

Request 를 활용한 메시지 확인

```
import requests
import mybot

def get_msg(msg):
    sender_id = msg['from']['id']
    sender_username = msg['from']['username']
    sender_fullname = msg['from']['first_name'] + ' ' + msg['from']['last_name']
    if 'text' in msg:
        text = msg['text']
        print(f'{sender_fullname}({sender_username}:{sender_id}) : {text}')

url = f'https://api.telegram.org/bot{mybot.token}/getUpdates'
r = requests.get(url)
r.json()
if r.ok:
    updates = r.json()['result']
    for u in updates:
        get_msg(u['message'])
```

읽은 메시지 무시하고 새로운 메시지만 읽기

```
# consume read message
if updates:
    last_update_id = updates[-1]['update_id']
    params = {
        'offset': last_update_id + 1
    }
    r = requests.get(url, params=params)
```

메시지 전송

```
def send_message(to_id, text):
    url = f'https://api.telegram.org/bot{mybot.token}/sendMessage'
    params = {
        'chat_id': to_id,
        'text': text
    }
    r = requests.get(url, params=params)
```

사진 (링크) 전송

```
def send_photo(to_id, link):
    url = f'https://api.telegram.org/bot{mybot.token}/sendPhoto'
    params = {
        'chat_id': to_id,
        'caption': '사진',
        'photo': link
    }
    r = requests.get(url, params=params)
    print(r.json())
    assert(r.ok)
```

문서 전송

```
def send_document(to_id, fname):
    url = f'https://api.telegram.org/bot{mybot.token}/sendDocument'
    params = {
        'chat_id': to_id,
        'caption': fname
    }
    with open(fname, 'rb') as f:
        files = { 'document' : f}
        r = requests.get(url, params=params, files=files)
        print(r.json())
        assert(r.ok)
```

봇에게 전송한 문서 파일

```
"document": {  
    "file_name": "\ubcbd\ub3cc\uc9d1 \uc2e4\uc2b5\uc2dc\ud5d8.pdf",  
    "mime_type": "application/pdf",  
    "thumbnail": {  
        "file_id": "AAMCBQADGQEAAw1kTe1cnWo9tFHMMjpaVSpQQISFUwACeQsAAtfnaFbJu_4DmRMBCQEAB20AAy8E",  
        "file_unique_id": "AQADeQsAAtfnaFZy",  
        "file_size": 11441,  
        "width": 227,  
        "height": 320  
    },  
    "thumb": {  
        "file_id": "AAMCBQADGQEAAw1kTe1cnWo9tFHMMjpaVSpQQISFUwACeQsAAtfnaFbJu_4DmRMBCQEAB20AAy8E",  
        "file_unique_id": "AQADeQsAAtfnaFZy",  
        "file_size": 11441,  
        "width": 227,  
        "height": 320  
    },  
    "file_id": "BQACAgUAAxkBAAMNZE3tXJ1qPbRRzDI6WIUqUEJUhVMAAnkLAALX52hWybv-A5kTAQkvBA",  
    "file_unique_id": "AgADeQsAAtfnaFY",  
    "file_size": 5618263  
},
```

file info 획득

```
def get_file_info(id):
    params = {
        'file_id': id
    }
    # first, get file info
    r = requests.get(f'https://api.telegram.org/bot{mybot.token}/getFile', params=params)
    if r.ok:
        return r.json()
    raise Exception('Telegram GetFile Failed')
```

```
{
    'ok': True,
    'result': {
        'file_id': 'AgACAgUAxkDAAMQZE7wckGQ5jOOZE5jUBPH3dHYKbcAAju4MRvX52hWEomY_JCZPeUBAAMCAAN5AAMvBA',
        'file_unique_id': 'AQADO7gxG9fnaFZ-',
        'file_size': 50670,
        'file_path': 'photos/file_0.jpg'
    }
}
```

문서 파일 다운 로드

```
def download_file(file_id, telegram_file_path, file_name):
    with open(file_name, 'wb') as wf:
        print(f'writing {file_name}.....')
        r = requests.get(f'https://api.telegram.org/file/bot{mybot.token}/' + telegram_file_path)
        if r.ok:
            wf.write(r.content)
        print('.....done')

def get_msg(msg):
    global sender_id
    sender_username = msg['from']['username']
    sender = msg['from']['first_name']

    if 'document' in msg:
        caption = msg['caption'] if 'caption' in msg else 'nocaption'
        file_name = msg['document']['file_name']
        file_id = msg['document']['file_id']
        file_info = get_file_info(file_id)
        telegram_file_path = file_info['result']['file_path']
        download_file(file_id, telegram_file_path, file_name)
```

봇에게 이미지 전송

```
"photo": [
  {
    "file_id": "AgACAgUAAxkBAAQAMZE3j7uaEBvJ9w9B4iWlch_pi8DUAJu4MRvX52hWEomY_JCZPeUBAAMCAANzAAMvBA",
    "file_unique_id": "AQADO7gxG9fnaFZ4",
    "file_size": 881,
    "width": 67,
    "height": 90
  },
  {
    "file_id": "AgACAgUAAxkBAAQAMZE3j7uaEBvJ9w9B4iWlch_pi8DUAJu4MRvX52hWEomY_JCZPeUBAAMCAANTAAMvBA",
    "file_unique_id": "AQADO7gxG9fnaFZy",
    "file_size": 8363,
    "width": 240,
    "height": 320
  },
  {
    "file_id": "AgACAgUAAxkBAAQAMZE3j7uaEBvJ9w9B4iWlch_pi8DUAJu4MRvX52hWEomY_JCZPeUBAAMCAAN4AAMvBA",
    "file_unique_id": "AQADO7gxG9fnaFZ9",
    "file_size": 29793,
    "width": 600,
    "height": 800
  },
  {
    "file_id": "AgACAgUAAxkBAAQAMZE3j7uaEBvJ9w9B4iWlch_pi8DUAJu4MRvX52hWEomY_JCZPeUBAAMCAAN5AAMvBA",
    "file_unique_id": "AQADO7gxG9fnaFZ-",
    "file_size": 50670,
    "width": 960,
    "height": 1280
  }
]
```

이미지 파일 다운로드

```
import uuid

def get_msg(msg):
    global sender_id
    sender_username = msg['from']['username']
    sender = msg['from']['first_name']

    if 'photo' in msg:
        caption = msg['caption'] if 'caption' in msg else 'NOCAPTION'
        file_id = msg['photo'][-1]['file_id']
        file_info = get_file_info(file_id)
        telegram_file_path = file_info['result']['file_path']
        file_name = f'img{uuid.uuid4().int}' + os.path.splitext(telegram_file_path)[-1]
        print(f'{sender}({sender_username}) : sent image "{file_name}" with {caption} @ {telegram_file_path}')
        download_file(file_id, telegram_file_path, file_name)
```

카카오톡 (나에게) 메시지 보내기

- Kakao REST API 사용
- <https://developers.kakao.com/docs/latest/ko/reference/rest-api-reference>
- 친구들에게 보내기 – 카카오 서비스팀의 인증이 필요.

The screenshot shows a browser window displaying the Kakao Developers REST API Reference. The URL in the address bar is <https://developers.kakao.com/docs/latest/ko/reference/rest-api-reference>. The page has a dark theme with a sidebar on the left containing links like '카카오 로그인', '카카오톡 소셜', '메시지', etc. The main content area features a large blue header with the text '유용한 참고 정보' (Useful Reference Information). Below this, there's a section titled 'REST API 레퍼런스' (REST API Reference) which includes a note about SSL and OAuth 2.0 authentication. At the bottom, there's a table for '카카오 로그인' (Kakao Login) with two rows:

URL	HOST	METHOD	기능
/oauth/authorize	https://auth.kakao.com	GET	인가 코드 받기 추가 승인 동의 받기 액션 선택에 동의 받기 독에서 자동 로그인
/oauth/token	https://kauth.kakao.com	POST	токен 받기, 토큰 갱신하기

메시지 보내기 절차

- Kakao developer 계정 만들기.
- 애플리케이션 추가. – REST API key 획득.
- 카카오 로그인 활성화. Redirect URI 설정. 등의 항목 설정
- 애플리케이션 인증 코드 발급.
- Access token 및 refresh token 발급.
- JSON 형식의 메시지 구성.
- Talk REST API 호출.

애플리케이션 추가

The screenshot shows the Kakao Developers console interface. At the top, there is a navigation bar with links for '내 애플리케이션' (My Applications), '제품' (Products), '문서' (Documentation), '도구' (Tools), '포럼' (Forum), and a user account section. Below the navigation bar, there is a search bar labeled '애플리케이션 이름' (Application Name). A large blue button with a white plus sign and the text '애플리케이션 추가하기' (Add Application) is prominently displayed. Below this button, three application cards are listed:

- PythonTalkToMe** (APP) - ID 579272, OWNER, Web
- SendMessageByPython** (APP) - ID 575235, OWNER
- 항공검색** (APP)

REST API 키 발급 확인

The screenshot shows a browser window for the Kakao Developers console at developers.kakao.com/console/app/579272/config/appKey. The page displays the configuration for the 'PythonTalkToMe' app, specifically the 'App Key' section.

Left Sidebar: Includes links for '앱 설정', '요약 정보', '일반', '비즈니스', '애키' (selected), '플랫폼', '팀 관리', '제품 설정', '카카오 로그인', '동의항목', '간편가입', and '카카오톡 채널'.

Header: Shows the 'kakao developers' logo, navigation links for '내 애플리케이션', '제품', '문서', '도구', '포럼', user info 'daehyun.lee@gmail.com', and language switch 'KOR ENG'.

Content Area: Displays the 'PythonTalkToMe' app details (ID 579272, OWNER, Web). The 'App Key' section lists four types of keys:

플랫폼	앱 키	재발급
네이티브 앱 키	e1ae921dbf8e7b684bd4b870cef04544	복사 재발급
REST API 키	671f9b487224f17e52fc538f17ef7f98	복사 재발급
JavaScript 키	0eb4db46cf9560d96eca3031c55f91e2	복사 재발급
Admin 키	1dcb1ffe39edccf43cac371d1f30264a	복사 재발급

Bottom Notes:

- 네이티브 앱 키: Android, iOS SDK에서 API를 호출할 때 사용합니다.
- JavaScript 키: JavaScript SDK에서 API를 호출할 때 사용합니다.

로그인 활성화 / Redirect URI 설정

The screenshot shows the Kakao Developers console interface. The URL in the address bar is `developers.kakao.com/console/app/579272/product/login`. The main content area displays the configuration for the app `PythonTalkToMe` (ID 579272, OWNER, Web). The left sidebar has a navigation menu with various settings like 플랫폼, 팀 관리, 제품 설정, 카카오 로그인, etc. The `카카오 로그인` section is currently selected and highlighted with a red box around its title. Inside this section, there are two main configuration areas: **활성화 설정** and **Redirect URI**. The **활성화 설정** area contains a switch labeled "ON" which is also highlighted with a red box. Below the switch, there is explanatory text about using Kakao API for user login. The **Redirect URI** area shows a field with the value `https://localhost.com`, which is also highlighted with a red box. There are "삭제" and "수정" buttons next to the Redirect URI field.

동의 항목 설정

■ 최소 설정: 프로필 정보, 메시지전송

The screenshot shows the Kakao Developers console interface. On the left, there's a sidebar with various settings like 플랫폼, 팀 관리, 제품 설정, 카카오 로그인, 동의항목, 간편가입, 카카오톡 채널, 개인정보 국외이전, 연결 끊기, 사용자 프로파티, 보안, 고급, 카카링크, 카카오톡 채널, 음성, 푸시 알림, 고급 설정, and 허용 ID 주소. A red box highlights the '동의항목' section under 카카오 로그인.

The main content area has two tabs: '동의항목' (selected) and '접근권한 관리'. Under '접근권한 관리', there's a table with columns '항목 이름', 'ID', and '상태'. It shows '카카오페이지 글 목록' with 'story_read' selected. Another red box highlights this row.

Under '동의항목', there's a table with columns '필수 동의' and '선택 동의'. It lists several items: '프로필 정보(닉네임/프로필 사진)' (profile), '카카오계정(이메일)' (account_email), '출생 연도' (birthyear), '카카오계정(전화번호)' (phone_number), 'CI(연계정보)' (account_ci), and '배송지정보(수령인명, 배송지 주소, 전화번호)' (shipping_address). The '선택 동의' column for '카카오페이지 글 목록' is checked.

At the bottom of the '동의항목' table, there's a note:

- 선택 동의: 사용자가 동의하지 않아도 카카오 로그인을 완료할 수 있습니다.
- 이용 중 동의: 카카오 로그인 시 동의를 받지 않고, 항목이 필요한 시점에 동의를 받습니다.
- 사용 안함: 사용자에게 동의를 요청하지 않습니다.

애플리케이션 인증 코드 발급

웹브라우저에서 아래의 주소로 접속해야 함.

```
https://kauth.kakao.com/oauth/authorize?client_id=xxxxxxxxxxxx&redirect_uri=https://localhost.com  
&response_type=code
```

webbrowser 모듈 이용

```
import webbrowser

url = "https://kauth.kakao.com/oauth/authorize"

params = {
    "client_id": kakao_account.REST_API_Key,
    "redirect_uri": "https://localhost.com",
    'response_type' : 'code'
}

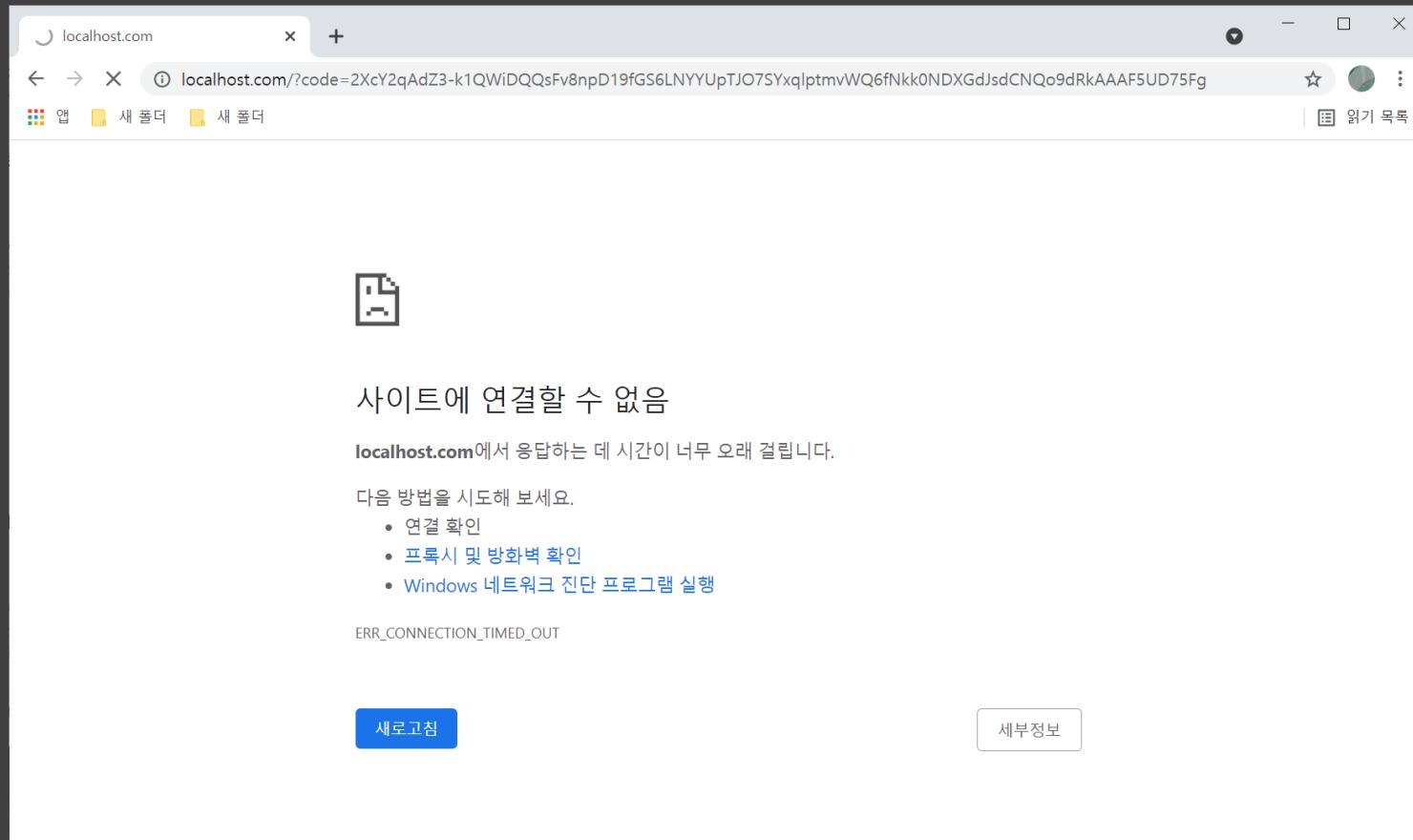
url_get_access_code = url + '?' + '&'.join([k+'=' + v for k,v in params.items()])
webbrowser.open(url_get_access_code)
```

The image shows a Windows desktop environment with three windows open:

- Top-left window:** A promotional window for Kakao account linking. It features a central smartphone icon with various Kakao service icons (Talk, Pay, D) connected to it by dashed lines. Text at the top reads "카카오계정 하나로 충분합니다." and "Kakao의 모든 서비스 뿐 아니라 Melon, Daum 등 다른 다양한 서비스까지 이제 카카오계정으로 이용해 보세요!"
- Top-right window:** The official Kakao login page. It has a large input field for email (daehyun.lee@gmail.com) and a password field (redacted). Below the fields are checkboxes for "로그인 상태 유지" and "로그인". A yellow "로그인" button is prominent. Below the main form are links for "회원가입", "QR코드 로그인", "카카오계정", and "비밀번호 찾기".
- Bottom-right window:** A secondary Kakao login window titled "카카오계정으로 로그인". It displays a QR code for "PythonTalkToMe" by "이대현". A yellow "전체 동의하기" button is highlighted with a checkmark. Below it is a detailed description of the terms of service.

At the bottom center of the desktop, there is a watermark or footer text: "Copyright © Kakao Corp. All rights reserved."

Redirect URI에서, code 값을 획득.



```
redirect_uri =  
'https://localhost.com/?code=GQCJs5or1zOkZBvfeg6zkmPPmQkVDFFIqngng7jYbFP22QfGphWY4DQrFZLMS6mzJfnnbUgorDNQAAAF5UG3H_Q'  
code = redirect_uri.split('=')[1]
```

Access token / refresh token 발급 및 저장

```
import json

url = "https://kauth.kakao.com/oauth/token"

data = {
    "grant_type": "authorization_code",
    "client_id": kakao_account.REST_API_Key,
    "redirect_uri": "https://localhost.com",
    "code": code
}

response = requests.post(url, data=data)

tokens = response.json()

print(f'Access token expires in {round(tokens["expires_in"]/3600)} hours.')
print(f'Refresh token expires in {round(tokens["refresh_token_expires_in"]/3600/24)} days.')

tokens['code'] = code
with open('kakao_tokens.json', 'w') as fp:
    json.dump(tokens, fp)
```

Access token 재발급 – refresh token 이용

```
url = "https://kauth.kakao.com/oauth/token"
data = {
    "grant_type" : "refresh_token",
    "client_id" : kakao_account.REST_API_Key,
    "refresh_token" : tokens['refresh_token']
}

response = requests.post(url, data=data)
tokens.update(response.json())

print(f'Access token expires in {round(tokens["expires_in"]/3600)} hours.')
print(f'Refresh token expires in {round(tokens["refresh_token_expires_in"]/3600/24)} days.')

with open('kakao_tokens.json', 'w') as fp:
    json.dump(tokens, fp)
```

메시지 구성 - <https://developers.kakao.com/docs/latest/ko/message/rest-api>

메시지 ^

- 이해하기
- 설정하기
- 메시지 템플릿
- 카카오링크: JavaScript
- 카카오링크: Android
- 카카오링크: iOS
- 카카오톡 메시지: REST API**
- 시작하기 전에
- 기본 템플릿으로 메시지 보내기
- 사용자 정의 템플릿으로 메시지 보내기
- 기본 템플릿으로 스크랩 메시지 보내기
- 사용자 정의 템플릿으로 스크랩 메시지 보내기
- 카카오톡 메시지: JavaScript

Request

URL

```
POST /v2/api/talk/memo/default/send HTTP/1.1
Host: kapi.kakao.com
Authorization: Bearer {ACCESS_TOKEN}
```

Parameter

Name	Type	Description	Required
template_object	Object	메시지 구성 요소를 담은 객체(Object) 피드, 리스트, 위치, 카머스, 텍스트 중 하나	O

Response

Key

Name	Type	Description
result_code	Integer	전송 성공 시 0

> Sample

Request

피드

리스트

위치

커머스

텍스트

```
curl -v -X POST "https://kapi.kakao.com/v2/api/talk/memo/default/send" \
-H "Authorization: Bearer {ACCESS_TOKEN}" \
-d 'template_object={
    "object_type": "text",
    "text": "텍스트 영역입니다. 최대 200자 표시 가능합니다.",
    "link": {
        "web_url": "https://developers.kakao.com",
        "mobile_web_url": "https://developers.kakao.com"
    },
    "button_title": "바로 확인"
}'
```

Response

```
HTTP/1.1 200 OK
{
    "result_code":0
}
```

```
import requests
import json

with open('kakao_tokens.json') as f:
    tokens = json.load(f)

url = "https://kapi.kakao.com/v2/api/talk/memo/default/send"

# headers 구성
headers = {
    "Authorization": "Bearer " + tokens['access_token']
}

data = {
    "template_object" : json.dumps(
        {
            "object_type" : "text",
            "text" : "파이썬에서 보내는 텍스트입니다.",
            "link" : {},
            "button_title" : "확인"
        }
    )
}
```

공백 포함!

```
response = requests.post(url, headers=headers, data=data)
print(response.status_code)
if response.json().get('result_code') == 0:
    print('메시지를 성공적으로 보냈습니다.')
else:
    print('메시지를 성공적으로 보내지 못했습니다. 오류메시지 : ' + str(response.json()))
```

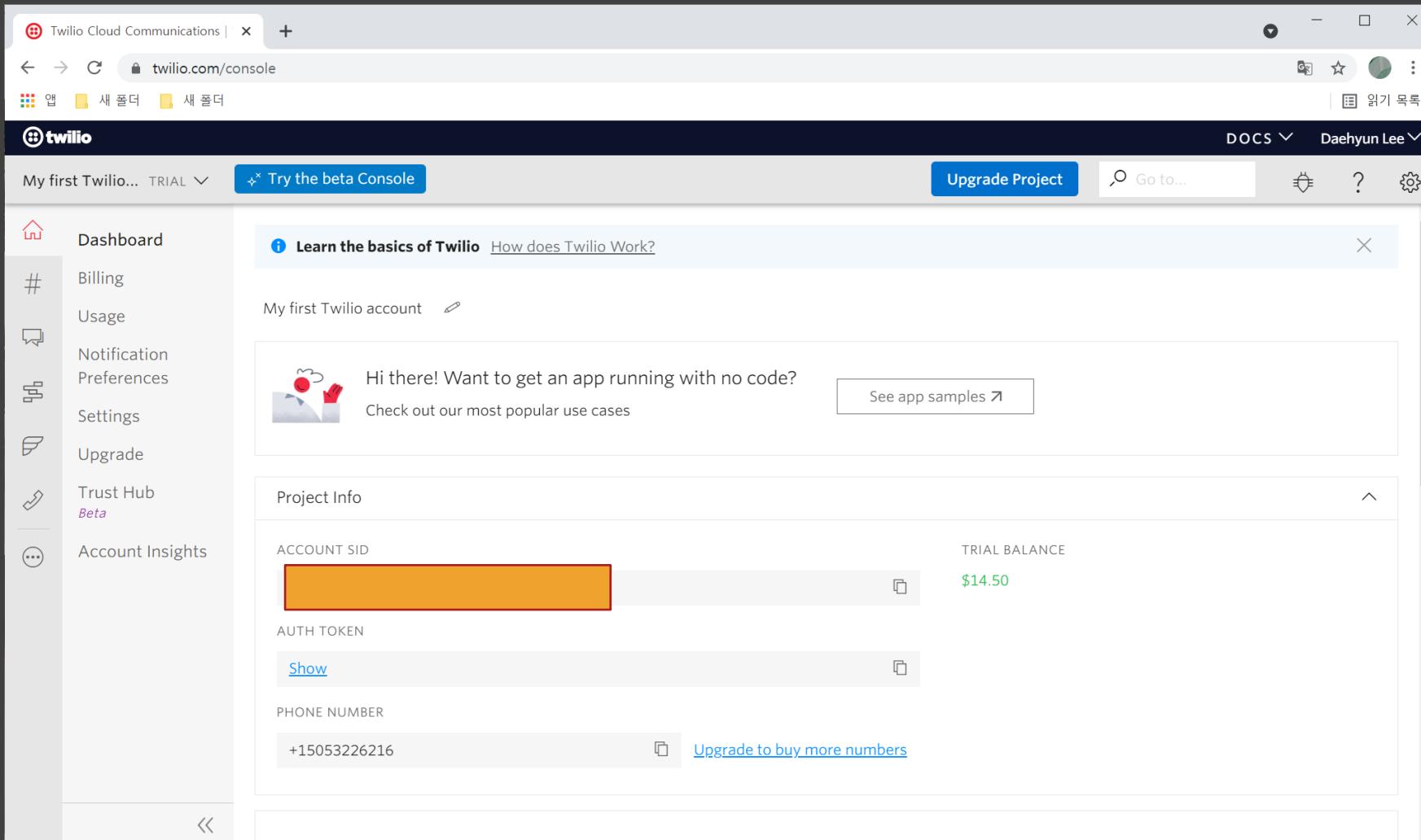
SMS 보내기

- **SMS gateway**
 - 이메일 보내듯이 메일을 보내면, 전화번호로 수신됨.
 - Ex. 미국 at&t 번호로 보내려면, 2223334444@txt.att.net 식으로..
 - 우리나라 통신사는 막아 놓았음.
 - **SMS 문자 발송 서비스(유로)**를 이용해야 함.
- 참고: 문자로 메일을 보낼 수 있음.

Mobile carrier	SMS gateway domain	MMS gateway domain
Alltel ^[9]	sms.alltelwireless.com	mmss.alltelwireless.com
AT&T ^[10]	txt.att.net	mmss.att.net
Boost Mobile ^[9]	sms.myboostmobile.com	myboostmobile.com
Cricket Wireless	mmss.cricketwireless.net	mmss.cricketwireless.net
FirstNet ^[9]	txt.att.net	mmss.att.net
Google Fi ^[11]		msg.fi.google.com
MetroPCS	mymetropcs.com	mymetropcs.com
Republic Wireless ^[12]	text.republicwireless.com	
Sprint ^[9]	messaging.sprintpcs.com	pm.sprint.com
T-Mobile ^[9]	tmomail.net	tmomail.net
U.S. Cellular ^[9]	email.uscc.net	mmss.uscc.net
Verizon Wireless ^[13]	vtext.com	vzwpix.com
Virgin Mobile ^[9]	vmobl.com	vmpix.com

Twilio

- 공짜로 백개 보낼 수 있음.



SMS API

The screenshot shows a browser window displaying the Twilio SMS API Overview page. The URL in the address bar is twilio.com/docs/sms/api. The page has a dark blue header with the Twilio logo, navigation links for Twilio Docs, SMS, Voice, Runtime, Video, Studio, All docs..., SDKs, Help, Search, and a Console link. On the left, there's a sidebar with a tree view of the API Reference, currently expanded to show the 'SMS API authentication' section. The main content area contains text about using the `frankfurt-ix` edge location and a code block showing the base URL. Below this, a section titled 'SMS API authentication' explains HTTP Basic authentication using Account SID and Auth Token. It includes a code block for a curl command and a note about finding these credentials in the Twilio Console. At the bottom, there's a link to learn more about authentication and interaction with the Twilio REST API.

If you have a private connection through [Interconnect](#) in Germany, you can make use of the `frankfurt-ix` edge location by using this base URL:

```
https://api.frankfurt-ix.us1.twilio.com/2010-04-01
```

SMS API authentication

HTTP requests to the API are protected with [HTTP Basic authentication](#). To learn more about how Twilio handles authentication, please refer to our [security documentation](#).

In short, you will use your Twilio **Account SID** as the username and your **Auth Token** as the password for HTTP Basic authentication with Twilio.

```
curl -G https://api.twilio.com/2010-04-01/Accounts \
-u <YOUR_ACCOUNT_SID>:<YOUR_AUTH_TOKEN>
```

You can find your Account SID and Auth Token in [the Console](#).

To learn more about authentication and interaction with the Twilio REST API, check out

twilio.rest 모듈 이용

```
from twilio.rest import Client
import twilio_account

# Your Account Sid and Auth Token from twilio.com/console
# and set the environment variables. See http://twilio.com/secure
account_sid = twilio_account.sid
auth_token = twilio_account.token
client = Client(account_sid, auth_token)

message = client.messages \
    .create(
        body="교수님. I LOVE YOU ~. 89418022 이대현",
        from_=twilio_account.phone_number,
        to='+821097xx17xx'
    )

print(message.sid)
```