

기획서

제목: Chatbot 을 이용한 가상화폐 시장 동향파악	기획자: 하창권
The idea Kakaotalk Chatbot 을 이용해서 가상화폐 시세와 관련 정보를 실시간으로 전달받고 빠르게 대응할 수 있다.	
Background <ul style="list-style-type: none">• 급변하는 가상화폐 시세• 가상화폐 거래소마다 다른 시세• 우수한 접근성을 가진 Application 의 부재	
How it works <ul style="list-style-type: none">• 누구에게: (Kakaotalk 을 사용하는) 가상화폐 시장에서 거래하는 사람에게• 언제: 원하는 시점이나 특정수치가 본인이 설정한 임계치에 도달했을 때• 어디서: (인터넷이 연결되는) 어디서든• 무엇을: 가상화폐의 시세파악, 가상화폐 구입판매, 가상화폐 관련 뉴스제공• 어떻게: Kakaotalk Chatbot 과의 대화를 통해	
Key benefits <ul style="list-style-type: none">• 본인이 설정한 변화폭이나 가격에 도달했을 경우 Kakaotalk 을 통해 사용자에게 알려줄 수 있어 시세에 민감한 사용자들에게 도움이 된다.• 가상화폐 거래소마다 다른 시세 파악을 위해 각각의 거래소 Application 을 여러 번 사용하는 것에 비해 Chatbot 은 한번에 시세를 파악할 수 있다.• Chatbot 에 입력한 키워드를 통해 가상화폐를 손쉽게 거래할 수 있다.• Kakaotalk 은 다른 어떤 Application 보다 접근성과 범용성이 보장된다.	
Next steps <ul style="list-style-type: none">• Kakaotalk Chatbot 제작하는 방법을 익힌다.(2017.11.17 까지)• 가상화폐 거래소의 API 를 분석하여 실현가능한 범위를 파악하고 추가적인 Idea 를 얻는다.(2017.11.22 까지)• 실력과 비용을 고려하여 실현가능한 범위를 명확히 한다.(2017.11.25 까지)• Chatbot 을 실제로 구현해 본다.(2017.11.30 까지)	

Chatbot을 이용한 가상화폐 시장 동향파악

제 프로젝트는 Django를 플랫폼으로 카카오톡 플러스친구
오픈소스와 빗썸과 코인원의 API를 활용한 챗봇입니다.

플러스친구 관리자

플러스친구 관리자센터

공지사항

플러스친구의 주소 정보가 개편됩니다. 주소 정보...

메시지 이용권0개 >

무료 발송 메시지10,000건

다른 플러스친구 선택하기

홈

메시지+

쿠폰+

1:1채팅+

스마트채팅

친구그룹 관리

홍보하기

통계+

관리+

이용가이드

공지사항>

고객센터

API형

별도의 개발의 통해 특정 답변을 요구하는 형태의 질문들을 설계하는 타입입니다.
API형 이용중 궁금한 점이 있으시면, [Github](#)으로 문의해주세요.

API Document

앱 등록

앱 이름changwon

앱 URLhttp://ec2-52-78-124-127.ap-northeast-2.compute.amazonaws.com:8000API 테스트

앱 설명챗봇실험

알림받을 전화번호 ?

☒ 플러스친구 API의 개인정보 수집 및 이용에 동의합니다.

전화번호대한민국(82)

카카오톡 이용중인 전화번호를 입력해주세요.

인증


대한민국(82)

010-3905-8245

삭제

카카오톡 플러스친구를 활용하였습니다.

플러스친구 Git_hub

 [plusfriend](#) / [auto_reply](#)


[Watch](#) 15 [Star](#) 175 [Fork](#) 66

[Code](#) [Issues](#) 97 [Pull requests](#) 3 [Projects](#) 0 [Wiki](#) [Insights](#)


플러스친구 자동응답 API

[20 commits](#) [1 branch](#) [0 releases](#) [6 contributors](#)

[Branch: master](#) [New pull request](#) [Create new file](#) [Upload files](#) [Find file](#) [Clone or download](#)

 [plusfriendteam](#) committed on [GitHub](#) Update README.md Latest commit d7ecf22 on 16 Aug

[README.md](#) Update README.md 4 months ago

 README.md

카카오톡 플러스친구 API v. 2.0 개요

이 문서는 플러스친구를 통하여 자동응답 기능을 이용하고자 할 때 사용되는 API에 대해 기술합니다.

1. 이용에 대한 참고사항

1. 서버간 통신은 보안을 위하여 HTTPS를 쓰도록 권장합니다.
2. HTTPS를 통하여 API를 이용하는 파트너사 서버는 유효한 공인인증서를 사용해야 합니다.
3. user_key는 외부에 노출되지 않도록 주의해 주시기 바랍니다.
4. 플러스친구 API 운영에 대한 정책 및 가이드라인은 [플러스친구 이용약관](#)의 API 플랫폼 운영정책을 참고해 주시기 바랍니다.

카카오톡 플러스친구 GIT_HUB를 활용하였습니다.

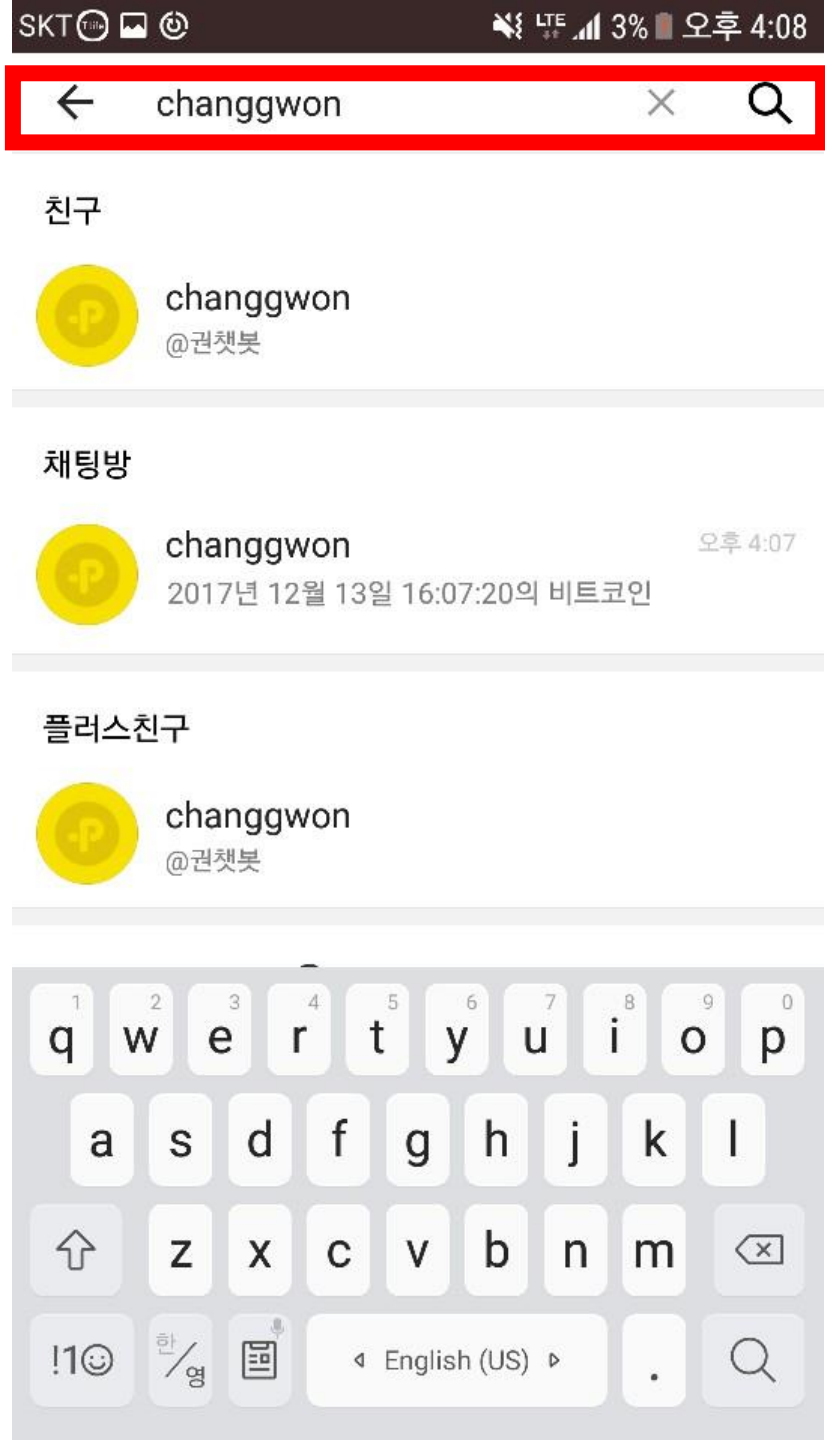
카카오톡 친구

검색창에

Changgwon 검색!

*2018.03.05 현재
실행되지 않습니다!

플러스친구를 통해 챗봇
을 구현한 모습입니다.



keyboard

Specification

- Method : GET
- URL : `http(s)://:your_server_url/keyboard`
- Content-Type : `application/json; charset=utf-8`
- 예제

```
curl -XGET 'https://:your_server_url/keyboard'
```

• Response

필드명	타입	필수여부	설명
keyboard	Keyboard	Required	키보드 영역에 표현될 버튼에 대한 정보. 생략시 <code>text</code> 타입이 선택된다.

• 예제

```
{
  "type" : "buttons",
  "buttons" : ["선택 1", "선택 2", "선택 3"]
}
```

카카오톡 플러스친구 API중 Keyboard API입니다.

message

- 예제

```
curl -XPOST 'https://your_server_url/message' -d '{  
  "user_key": "encryptedUserKey",  
  "type": "text",  
  "content": "차량번호등록"  
}'
```

```
curl -XPOST 'https://your_server_url/message' -d '{  
  "user_key": "encryptedUserKey",  
  "type": "photo",  
  "content": "http://photo_url/number.jpg"  
}'
```

- Response

필드명	타입	필수여부	설명
message	Message	Required	자동응답 명령어에 대한 응답 메시지의 내용. 6.2에서 상세 기술
keyboard	Keyboard	Optional	키보드 영역에 표현될 명령어 버튼에 대한 정보. 생략시 text 타입(주관식 답변 키보드)이 선택된다. 6.1에서 상세 기술

- 예제

```
{  
  "message": {  
    "text": "귀하의 차량이 성공적으로 등록되었습니다. 축하합니다!"  
  }  
}
```

카카오톡 플러스친구 API중 message API입니다.

urls.py

```
from django.conf.urls import url
from soganghaksik import views
```

```
urlpatterns = [
    url(r'^keyboard/', views.keyboard),
    url(r'^message', views.answer),
]
```

Python 코드로 keyboard와 message를 코딩한 모습입니다.

첫 화면 (views)

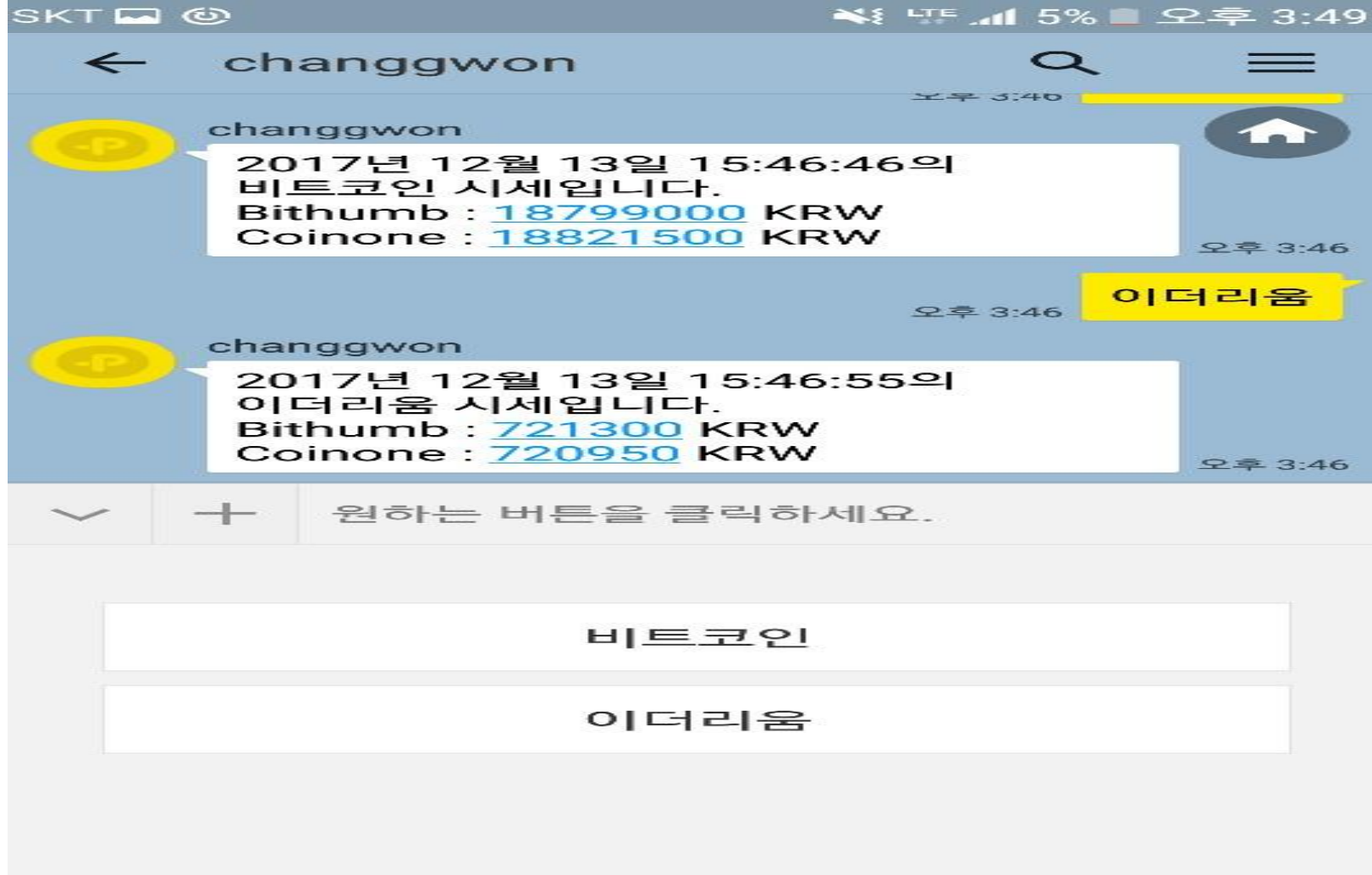
```
def keyboard(request):  
  
    return JsonResponse({  
        'type' : 'buttons',  
        'buttons' : ['비트코인', '이더리움']  
    })
```

Python 코드로 Keyboard를 코딩한 모습입니다.

버튼 클릭 (views)

```
def answer(request):  
  
    json_str = ((request.body).decode('utf-8'))  
    received_json_data = json.loads(json_str)  
    name = received_json_data['content']  
    today_date = datetime.datetime.now()  
    utcnow = datetime.datetime.utcnow()  
    time_gap = datetime.timedelta(hours=9)  
    kor_time = (utcnow + time_gap).strftime("%Y년 %m월 %d일 %H:%M:%S")  
  
  
    return JsonResponse({  
  
        'message': {'text': kor_time + '의 ' + name + ' 시세입니다.' + get_coin(name)},  
        },  
        'keyboard': {  
            'type': 'buttons',  
            'buttons': ['비트코인', '이더리움']  
        }  
    })
```

Python 코드로 message를 코딩한 모습입니다.



실제로 구현된 keyboard와 message의 모습입니다.

Bithumb_API (가격)

1. Public API

Resource	Description
----------	-------------

<https://api.bithumb.com/public/ticker/{currency}> [더보기](#)

bithumb 거래소 마지막 거래 정보

* {currency} = BTC, ETH, DASH, LTC, ETC, XRP, BCH, XMR, ZEC, QTUM, BTG (기본값: BTC), ALL(전체)

[Returned Example]

```
1  {
2      "status": "0000",
3      "data": {
4          "opening_price": "504000",
5          "closing_price": "505000",
6          "min_price": "504000",
7          "max_price": "516000",
8          "average_price": "509533.3333",
9          "units_traded": "14.71960286",
10         "volume_1day": "14.71960286",
11         "volume_7day": "15.81960286",
12         "buy_price": "505000",
13         "sell_price": "504000",
14         "date": "1417141032622"
15     }
16 }
```

[Returned Value Description]

Key Name	Description
status	결과 상태 코드 (정상 : 0000, 정상이외 코드는 에러 코드 참조)
opening_price	최근 24시간 내 시작 거래금액
closing_price	최근 24시간 내 마지막 거래금액
min_price	최근 24시간 내 최저 거래금액
max_price	최근 24시간 내 최고 거래금액
average_price	최근 24시간 내 평균 거래금액
units_traded	최근 24시간 내 Currency 거래량
volume_1day	최근 1일간 Currency 거래량
volume_7day	최근 7일간 Currency 거래량
buy_price	거래 대기건 최고 구매가
sell_price	거래 대기건 최소 판매가
date	현재 시간 Timestamp

Bithumb의 API중 가상화폐 가격 API입니다.

crawl method (views)

```
def crawl():
```

```
    coin_db = Coin.objects.all()  
    coin_db.delete()
```

DB 비우기

```
    urlTicker_B = urlopen('https://api.bithumb.com/public/ticker/all')  
    readTicker_B = urlTicker_B.read()  
    urlTicker_B.close()  
    jsonTicker_B = json.loads(readTicker_B.decode('utf-8'))
```

```
    FindBTC = jsonTicker_B['data']['BTC']['closing_price']  
    BTC_B = int(FindBTC)  
    FindETH = jsonTicker_B['data']['ETH']['closing_price']  
    ETH_B = int(FindETH)  
    FindDASH = jsonTicker_B['data']['DASH']['closing_price']  
    DASH_B = int(FindDASH)
```

```
    urlTicker_C = urlopen('https://api.coinone.co.kr/ticker/?currency=all')  
    readTicker_C = urlTicker_C.read()  
    jsonTicker_C = json.loads(readTicker_C.decode('utf-8'))  
    FindETC = jsonTicker_C['etc']['last']  
    ETC_C = int(FindETC)  
    FindBTC = jsonTicker_C['btc']['last']  
    BTC_C = int(FindBTC)  
    FindETH = jsonTicker_C['eth']['last']  
    ETH_C = int(FindETH)
```

```
    create_coin_db('비트코인',BTC_B,BTC_C)  
    create_coin_db('이더리움',ETH_B,ETH_C)
```

DB 채우기

빗썸과 코인원에서 가상화폐 가격을 가져와 DB에 채우는 코드입니다.

models (DB 구조)

```
16 lines (8 sloc) | 318 Bytes
Raw Blame History
1  from django.db import models
2
3  # Create your models here.
4
5
6  class Coin(models.Model):
7
8
9
10     coin_name = models.CharField(max_length=30, default="", primary_key=True)
11     bithumb_price = models.IntegerField(default=0)
12     coinone_price = models.IntegerField(default=0)
13
14     def __str__(self):
15         return self.title
```

```
def create_coin_db(coin_name, bithumb_price, coinone_price):
    Coin(coin_name=coin_name, bithumb_price=bithumb_price, coinone_price=coinone_price).save()
```

데이터베이스의 구조입니다.

get_coin method (views)

```
def get_coin(name):  
    crawl()  
    if name == '비트코인':  
        try:  
            a = Coin.objects.get(coin_name='비트코인').bithumb_price  
            b = Coin.objects.get(coin_name='비트코인').coinone_price  
  
        except Coin.DoesNotExist:  
            return "None"  
  
        return "\n" + "Bithumb : " + str(a) + " KRW" + "\n" + "Coinone : " + str(b) + " KRW"  
  
    else:  
        try:  
            a = Coin.objects.get(coin_name='이더리움').bithumb_price  
            b = Coin.objects.get(coin_name='이더리움').coinone_price  
        except Coin.DoesNotExist:  
            return "None"  
  
        return "\n" + "Bithumb : " + str(a) + " KRW" + "\n" + "Coinone : " + str(b) + " KRW"
```

DB에서 가상화폐 가격을 가져와 message로 출력하는 코드입니다.



실제로 구현된 Message입니다.

기획한 것에 비해 많이 초라한 결과물입니다!
멋사에서 더 많이 배워서 꼭 완성하고 싶습니다.
감사합니다!