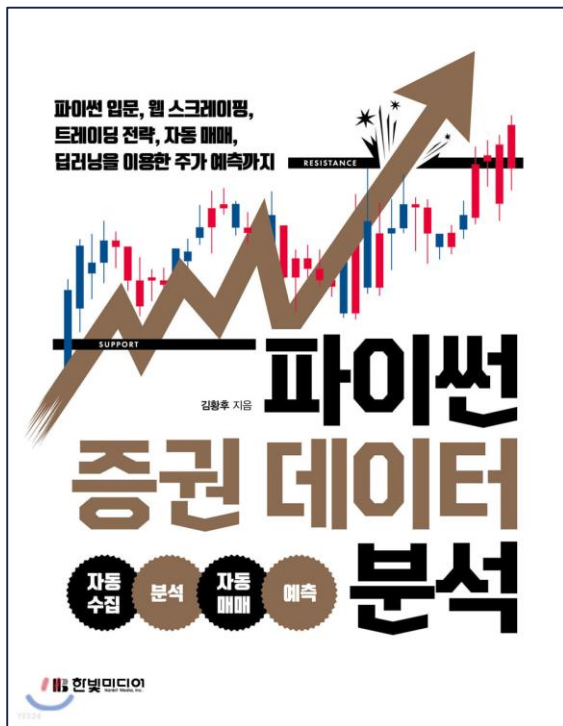


## CUAI 스터디 금융AI 1팀

2022.05.03

발표자 : 김진재

## 스터디 교재 / 모임 일시

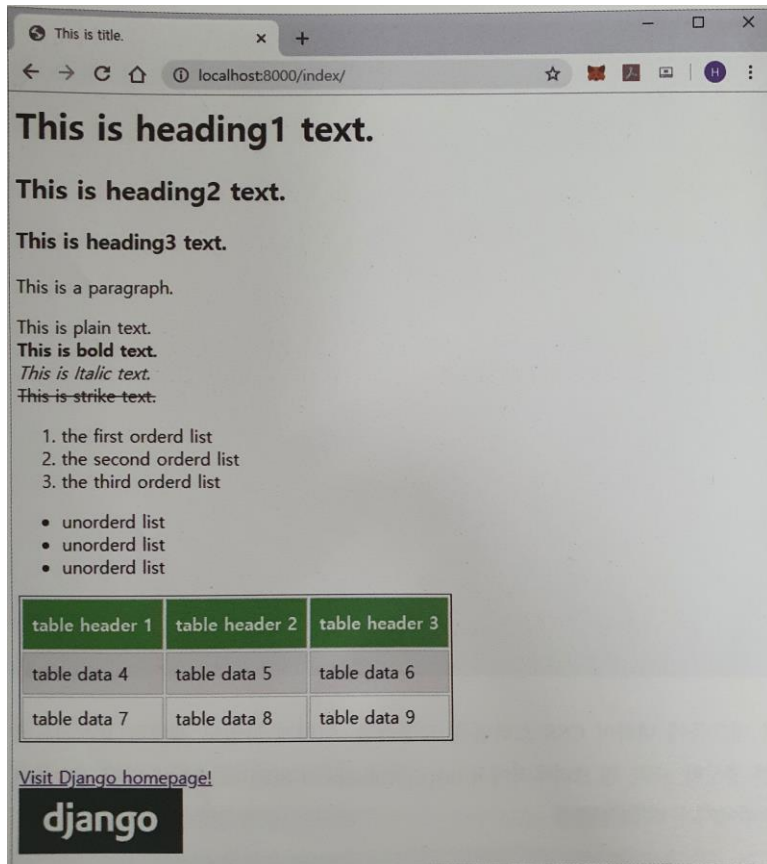


주차	날짜	내용
5	03/30	7장 : 장고 웹 서버 구축 및 자동화
6	04/07	6장 : 트레이딩 전략과 구현

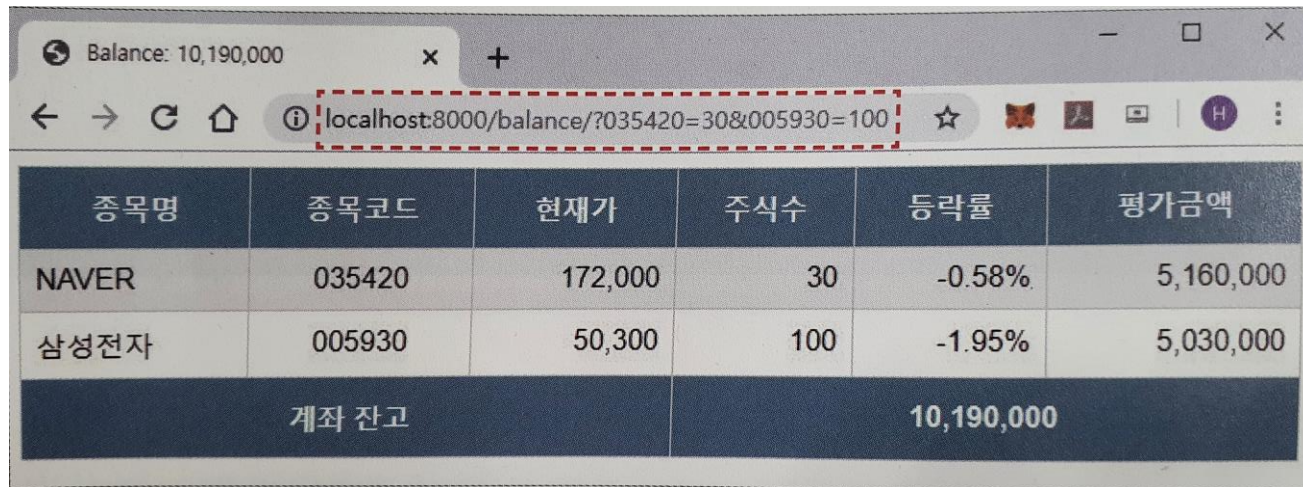
## 7장; 장고 웹 서버 구축 및 자동화

django

## 7장: 장고 웹 서버 구축 및 자동화



## 7장: 웹으로 계좌 잔고 확인하기



종목명	종목코드	현재가	주식수	등락률	평가금액
NAVER	035420	172,000	30	-0.58%	5,160,000
삼성전자	005930	50,300	100	-1.95%	5,030,000
계좌 잔고			10,190,000		

## 7장: 웹으로 계좌 잔고 확인하기

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
    'hello',  
    'index',  
    'balance', # ①  
]
```

```
from balance import views as balance_views # ②  
  
urlpatterns = [  
    path('admin/', admin.site.urls),  
    re_path(r'^(?P<name>[A-Z][a-z]*)$', views.sayHello),  
    path('index/', index_views.main_view),  
    path('balance/', balance_views.main_view), # ③  
]
```



# THO

```
# ch07_04_balance_views.py
from django.shortcuts import render
from bs4 import BeautifulSoup
import requests

def get_data(symbol):
    url = 'http://finance.naver.com/item/sise.nhn?code={}'.format(symbol)
    html = requests.get(url, headers={'User-agent': 'Mozilla/5.0'}).text
    soup = BeautifulSoup(html, "lxml", from_encoding="euc-kr")
    cur_price = soup.find('strong', id='nowVal') # ㉑
    cur_rate = soup.find('strong', id='rate') # ㉒
    stock = soup.find('title') # ㉓
    stock_name = stock.text.split(':')[0].strip() # ㉔
    return cur_price.text, cur_rate.text.strip(), stock_name

def main_view(request):
    querydict = request.GET.copy()
    mylist = querydict.lists() # ㉕
    rows = []
    total = 0

    for x in mylist:
        cur_price, cur_rate, stock_name = get_data(x[0]) # ㉖
        price = cur_price.replace(',', '')
        stock_count = format(int(x[1][0]), ',') # ㉗
        sum = int(price) * int(x[1][0])
        stock_sum = format(sum, ',')
        rows.append([stock_name, x[0], cur_price, stock_count, cur_rate,
                     stock_sum]) # ㉘
        total = total + int(price) * int(x[1][0]) # ㉙

    total_amount = format(total, ',')
    values = {'rows': rows, 'total': total_amount} # ㉚
    return render(request, 'balance.html', values) # ㉛
```

## 7장: 웹으로 계좌 잔고 확인하기

Balance: 50,575,000

localhost:8000/balance/?035420=30&005930=10...

종목명	종목코드	현재가	주식수	등락률	평가금액
NAVER	035420	172,000	30	-0.58%	5,160,000
삼성전자	005930	50,300	100	-1.95%	5,030,000
셀트리온	068270	174,000	50	-2.52%	8,700,000
카카오	035720	155,500	40	-0.96%	6,220,000
TIGER 미국채10년선물	305080	11,575	2,200	+0.17%	25,465,000
계좌 잔고			50,575,000		



## 7장: 슬랙으로 알림 메시지 보내기



## 7장; 백트레이더를 활용한 백테스트(RSI)

$RS = N\text{일간의 상승폭 평균} / N\text{일간의 하락폭 평균}$

$RSI = 100 - 100/(1+RS)$

## 7장: 백트레이더를 활용한 백테스트(RSI)

```
# ch07_08_Backtrader_RSI.py
from datetime import datetime
import backtrader as bt

class MyStrategy(bt.Strategy): # ①
    def __init__(self):
        self.rsi = bt.Indicators.RSI(self.data.close) # ②
    def next(self): # ③
        if not self.position:
            if self.rsi < 30:
                self.order = self.buy()
        else:
            if self.rsi > 70:
                self.order = self.sell()

cerebro = bt.Cerebro() # ④
cerebro.addstrategy(MyStrategy)
data = bt.feeds.YahooFinanceData(dataname='036570.KS', # ⑤
                                fromdate=datetime(2017, 1, 1), todate=datetime(2019, 12, 1))
cerebro.adddata(data)
cerebro.broker.setcash(10000000) # ⑥
cerebro.addsizer(bt.sizers.SizerFix, stake=30) # ⑦

print(f'Initial Portfolio Value : {cerebro.broker.getvalue():.0f} KRW')
cerebro.run() # ⑧
print(f'Final Portfolio Value : {cerebro.broker.getvalue():.0f} KRW')
cerebro.plot() # ⑨
```

Initial Portfolio Value : 10,000,000 KRW

Final Portfolio Value : 12,925,000 KRW



## 7장: 백트레이더를 활용한 백테스트(RSI\_SMA)

```
# ch07_09_Backtrader_RSI_SMA.py
import backtrader as bt
from datetime import datetime

class MyStrategy(bt.Strategy):
    def __init__(self):
        self.dataclose = self.datas[0].close
        self.order = None
        self.buyprice = None
        self.buycomm = None
        self.rsi = bt.indicators.RSI_SMA(self.data.close, period=21)

    def notify_order(self, order): # ①
        if order.status in [order.Submitted, order.Accepted]:
            return
        if order.status in [order.Completed]: # ②
            if order.isbuy():
                self.log(f'BUY : 추가 {order.executed.price:,.0f}, '
                        f'수량 {order.executed.size:,.0f}, '
                        f'수수료 {order.executed.comm:,.0f}, '
                        f'자산 {cerebro.broker.getvalue():,.0f}')
                self.buyprice = order.executed.price
                self.buycomm = order.executed.comm
            else:
                self.log(f'SELL : 추가 {order.executed.price:,.0f}, '
                        f'수량 {order.executed.size:,.0f}, '
                        f'수수료 {order.executed.comm:,.0f}, '
                        f'자산 {cerebro.broker.getvalue():,.0f}')
            self.bar_executed = len(self)
        elif order.status in [order.Canceled]:
            self.log('ORDER CANCELED')
        elif order.status in [order.Margin]:
            self.log('ORDER MARGIN')
```

```
elif order.status in [order.Rejected]:
    self.log('ORDER REJECTED')
    self.order = None

def next(self):
    if not self.position:
        if self.rsi < 30:
            self.order = self.buy()
    else:
        if self.rsi > 70:
            self.order = self.sell()

def log(self, txt, dt=None): # ③
    dt = self.datas[0].datetime.date(0)
    print(f'[{dt.isoformat()}] {txt}')

cerebro = bt.Cerebro()
cerebro.addstrategy(MyStrategy)
data = bt.feeds.YahooFinanceData(dataname='036570.KS',
                                  fromdate=datetime(2017, 1, 1), todate=datetime(2019, 12, 1))
cerebro.adddata(data)
cerebro.broker.setcash(10000000)
cerebro.broker.setcommission(commission=0.0014) # ④
cerebro.addsizer(bt.sizers.PercentSizer, percents=90) # ⑤

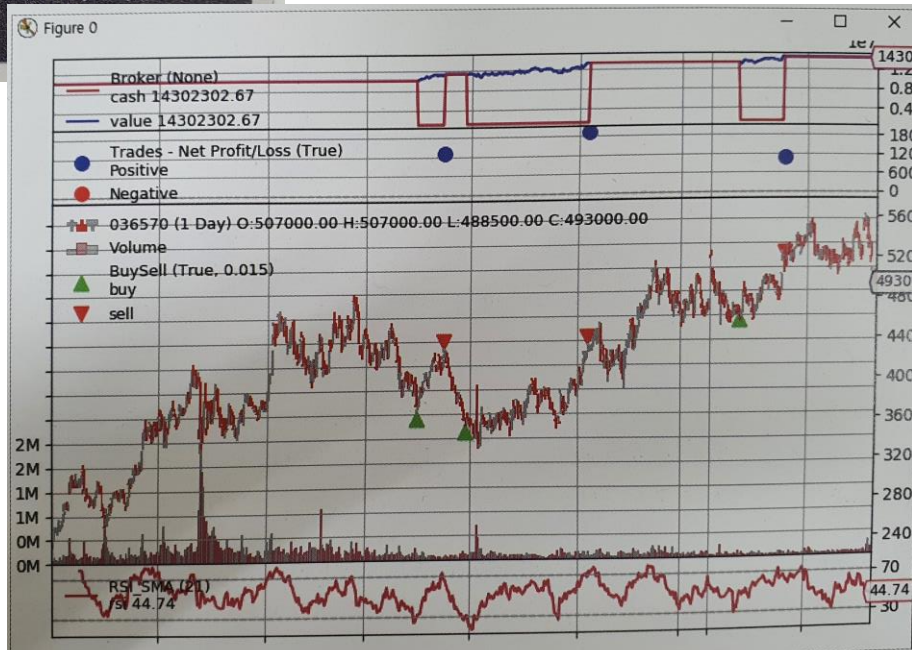
print(f'Initial Portfolio Value : {cerebro.broker.getvalue():,.0f} KRW')
cerebro.run()
print(f'Final Portfolio Value : {cerebro.broker.getvalue():,.0f} KRW')
cerebro.plot(style='candlestick') # ⑥
```



## 7장: 백트레이더를 활용한 백테스트(RSI\_SMA)

```
Python 3.8.0 Shell
File Edit Shell Debug Options Window Help

===== RESTART: C:/myPackage/Backtrader_RSISMA.py =====
Initial Portfolio Value : 10,000,000 KRW
[2018-03-02] BUY : 주가 370,000, 수량 24, 수수료 12,432, 자산 10,083,568
[2018-04-02] SELL : 주가 425,000, 수량 -24, 수수료 14,280, 자산 11,293,288
[2018-04-25] BUY : 주가 358,000, 수량 28, 수수료 14,150, 자산 11,166,205
[2018-09-13] SELL : 주가 427,000, 수량 -28, 수수료 16,878, 자산 13,210,352
[2019-06-17] BUY : 주가 470,500, 수량 25, 수수료 16,680, 자산 13,143,025
[2019-08-06] SELL : 주가 515,000, 수량 -25, 수수료 18,258, 자산 14,302,303
Final Portfolio Value : 14,302,303 KRW
>>>
```





## 감사합니다

CUAI 스터디 금융AI 1팀

발표자 : 김진재