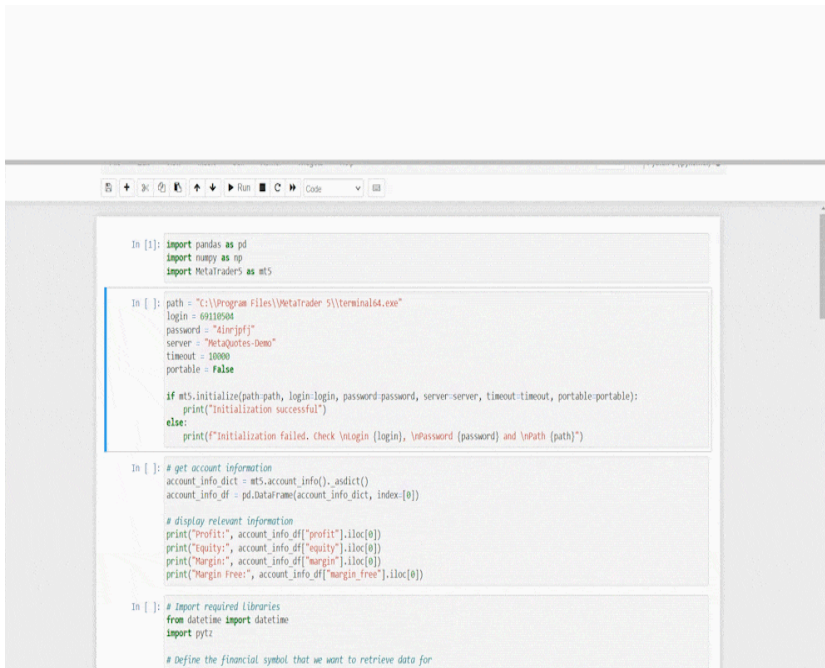


AUTOMATED TRADING USING MT5 AND PYTHON

Python and MT5 are great tools for automating your trading strategies. Python has powerful libraries for analysing data and developing trading strategies, while MT5 supports [automated trading](#) with Expert Advisors and other tools. By combining the two, you can retrieve data, generate signals, and place orders automatically. This can save you time and improve your trading performance.



```
In [1]: import pandas as pd
import numpy as np
import MetaTrader5 as mt5

In [2]: path = "C:\\Program Files\\MetaTrader 5\\terminal64.exe"
login = 69110594
password = "41njp1fj"
server = "MetaQuotes-Demo"
timeout = 10000
portable = False

if mt5.initialize(path=path, login=login, password=password, server=server, timeout=timeout, portable=portable):
    print("Initialization successful")
else:
    print(f"Initialization failed. Check \\nlogin ({login}), \\npassword ({password}) and \\npath ({path})")

In [3]: # get account information
account_info_dict = mt5.account_info().asdict()
account_info_df = pd.DataFrame(account_info_dict, index=[0])

# display relevant information
print("Profit:", account_info_df["profit"].iloc[0])
print("Equity:", account_info_df["equity"].iloc[0])
print("Margin:", account_info_df["margin"].iloc[0])
print("Margin Free:", account_info_df["margin_free"].iloc[0])

In [4]: # Import required libraries
from datetime import datetime
import pytz

# Define the financial symbol that we want to retrieve data for
symbol = "EURUSD"
```

RELATED KEYWORDS

- [Automated Trading on Alpaca: Step-by-Step Guide](#)
- [How to Backtest Trading Strategies in Pine Script](#)

Steps to Automate Trading Strategy using MT5 and Python

Step 1: Installation

*Please note that the following steps are specific to **Windows operating system**, as there is currently **no compatible MetaTrader5 Python package for other operating systems**. If you are using an operating system other than Windows, you may consider running these steps on a cloud platform such as Amazon Web Services.*

To get started on Windows OS, you need to install two things: the MT5 platform and a Jupyter notebook with the MetaTrader5

easily install it by following the steps outlined in this blog post:
<https://blog.quantinsti.com/set-up-python-system/>.

- Once you have Python installed, you can open a Jupyter notebook and install the MetaTrader5 package by typing the following command. This will allow you to connect to the MT5 platform from Python:

!pip install MetaTrader5

```
Collecting MetaTrader5Note: you may need to restart the kernel to use updated packages.

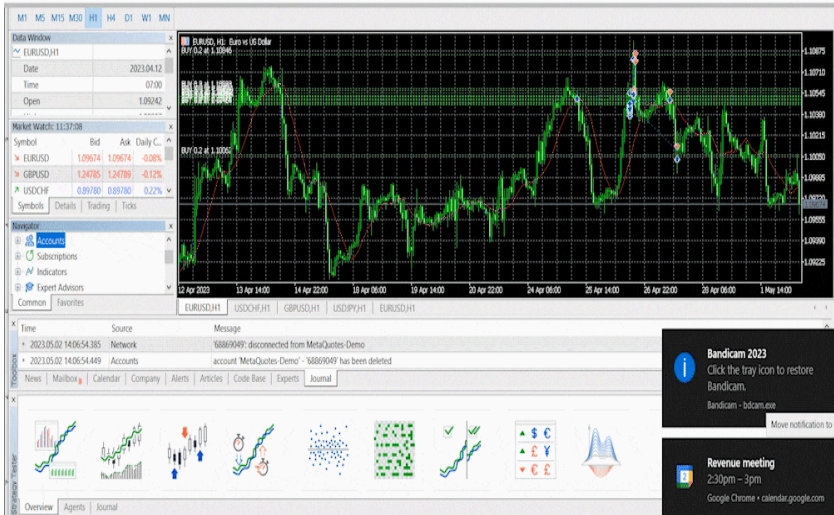
  Downloading MetaTrader5-5.0.45-cp39-cp39-win_amd64.whl (57 kB)
    ----- 0.0/57.2 kB ? eta -:--:--
    ----- 20.5/57.2 kB 682.7 kB/s eta 0:00:01
    ----- 57.2/57.2 kB 601.5 kB/s eta 0:00:00
Requirement already satisfied: numpy>=1.7 in c:\users\academy\miniconda3\lib\site-packages (from MetaTrader5) (1.24.2)
Installing collected packages: MetaTrader5
Successfully installed MetaTrader5-5.0.45
```

Step 2: Open a Demo Account & Get Login Credentials

To start, open the MetaTrader5 platform on your desktop. Once it's open, you'll need to follow a few simple steps to open a demo account:

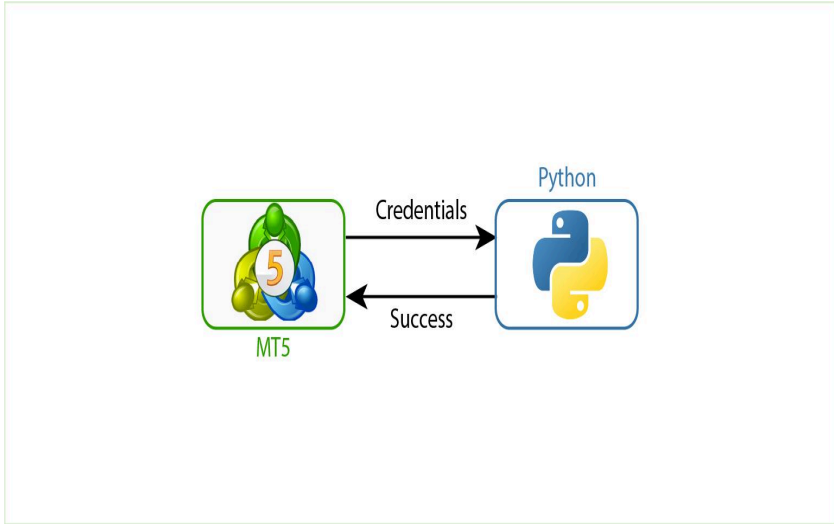
- In the "Navigator" window on the left-hand side, click on the "Accounts" tab.
- In the "Accounts" tab, click on the "MetaQuotes-Demo" server.
- Click the "Open an Account" button in the bottom left corner of the window.
- In the "Open an Account" dialog box, select "MetaQuotes-Demo" as the server and click "Next".
- Fill in the required personal information and click "Next".
- Choose the account type and currency and click "Next".
- Choose your leverage and deposit amount and click "Next".

cookies. You can read more about our privacy policy here



And that's it! Your login and password for the MetaQuotes-Demo server will be displayed in the "Accounts" tab of the "Navigator" window. The MetaQuotes-Demo server is for demo/training purposes only and does not use real money. Any profits or losses made on this server are not real and do not affect your real trading account.

Step 3: Initialize Connection



To start, you'll need to copy your login account number and password from the previous step. You'll also need to locate the `terminal64.exe` file, which is usually located in the `C:\\Program Files\\MetaTrader 5\\terminal64.exe` folder for Windows users.

Once you have these details, you need to set the server to `MetaQuotes-Demo`. This tells the MetaTrader 5 platform which server to connect to. To initialize your connection, you can use the following code:

```
import MetaTrader5 as mt5
import pandas as pd
```

cookies. You can read more about our privacy policy [here](#)

```
password = < your Password Here >
server = "MetaQuotes-Demo"
timeout = 10000
portable = False
```

```
if mt5.initialize(path=path, login=login, password=password,
server=server, timeout=timeout, portable=portable):
    print("Initialization successful")
```

Once you run this code, you should see a message indicating whether the initialization was successful or not. If it was successful, you're ready to move on to the next step! If not, double-check your login, password, and path to make sure everything is entered correctly.

Step 4: Account Information

You can find important information about your account using the `account_info` function. It will give you all the details you need to know, like your profit, equity, margin, and margin free.

```
# get account information
account_info_dict = mt5.account_info()._asdict()
account_info_df = pd.DataFrame(account_info_dict, index=[0])

# display relevant information
print("Profit:", account_info_df["profit"].iloc[0])
print("Equity:", account_info_df["equity"].iloc[0])
print("Margin:", account_info_df["margin"].iloc[0])
print("Margin Free:", account_info_df["margin_free"].iloc[0])
```

Step 5.1: Retrieve Hourly Data

You can choose the timeframe you want, like hourly or minute, and specify how far back you want to retrieve the data. Then, we convert the data into a Pandas DataFrame, which makes it easy to work with in Python.

```
timeframe = mt5.TIMEFRAME_H1
```

In this example, we are using the EURUSD currency pair and hourly time frame.

- Next, you need to set the end date and time before which you want to retrieve the data. You can use the datetime module to set the end time to the current time. MT5 takes input in UTC time. Therefore, we are converting the current time to UTC.

```
from datetime import datetime
import pytz
end_time = datetime.today().astimezone(pytz.utc)
```

- Now you can retrieve the historical data using the copy_rates_from function:

```
eurusd_rates = mt5.copy_rates_from(symbol, timeframe,
end_time, 10)
```

This retrieves the 10 candles of hourly EURUSD data before the end time.

- We convert the data to a Pandas DataFrame:

```
eurusd_df = pd.DataFrame(eurusd_rates)
```

	time	open	high	low	close	tick_volume	spread	real_volume
0	1682380800	1.10457	1.10479	1.10406	1.10460	1091	9	0
1	1682384400	1.10459	1.10519	1.10456	1.10509	509	1	0
2	1682388000	1.10512	1.10571	1.10494	1.10570	807	0	0

- To convert Unix timestamps to datetime objects, we use the "pd.to_datetime" function with two arguments: the column containing the Unix timestamps and the unit of time. Unix timestamp represents time as seconds since January 1, 1970, at 00:00:00 UTC.

```
eurusd_df['time'] = pd.to_datetime(eurusd_df['time'],
unit='s').dt.tz_localize('UTC')
```

We are using the "dt.tz_localize" function to set the timezone of datetime objects to UTC. This is because the Unix timestamps from MT5 are in UTC.

cookies. You can read more about our privacy policy here
in a different timezone than UTC.

```
eurusd_df['time'] = eurusd_df['time'].dt.tz_convert('US/Eastern')
```

	time	open	high	low	close	tick_volume	spread	real_volume
0	2023-04-24 20:00:00-04:00	1.10457	1.10479	1.10406	1.10460	1091	9	0
1	2023-04-24 21:00:00-04:00	1.10459	1.10519	1.10456	1.10509	509	1	0
2	2023-04-24 22:00:00-04:00	1.10512	1.10571	1.10494	1.10570	807	0	0

Step 5.2: Get Tick Data

This code retrieves tick data for the EUR/AUD currency pair, 20 ticks before the end_time. The mt5.copy_ticks_from() function is used to retrieve the tick data, and the resulting data is converted to a pandas DataFrame using pd.DataFrame().

```
euraud_tick = mt5.copy_ticks_from("EURAUD", end_time, 20, mt5.COPY_TICKS_ALL)
euraud_tick = pd.DataFrame(euraud_tick)
euraud_tick['time'] = pd.to_datetime(euraud_tick['time'], unit='s')
```

	time	bid	ask	last	volume	time_msc	flags	volume_real
0	2023-04-25 09:00:04	1.65470	1.65477	0.0	0	1682413204106	6	0.0
1	2023-04-25 09:00:04	1.65475	1.65482	0.0	0	1682413204202	6	0.0
2	2023-04-25 09:00:06	1.65477	1.65484	0.0	0	1682413206913	6	0.0
3	2023-04-25 09:00:07	1.65480	1.65487	0.0	0	1682413207424	6	0.0

time_msc: The timestamp of the tick data in milliseconds

You can replicate the steps on your local computer and experiment with different timeframes by adjusting the parameter timeframe. For example, to obtain data for 1-minute intervals, update the code with timeframe = mt5.TIMEFRAME_M1, or [choose any other desired timeframe](#).

Step 6: Trading Signals

To generate trading signals, we need to analyze the data we've collected from the MT5. This can be done using different

cookies. You can read more about our privacy policy [here](#)

to perform well in the future, and vice versa. On the other hand, mean reversion strategies rely on the assumption that the stocks will eventually return to their average price, after deviating from it.

We can also use popular technical indicators such as moving averages, Bollinger bands, and Relative Strength Index (RSI) to generate trading signals. These indicators help us to identify trends, overbought and oversold conditions, and potential entry and exit points.

If you're interested in learning more about these strategies and other trading concepts, Quantra is a great resource. They offer over 50 courses on quantitative and algo trading, covering everything from beginner concepts to advanced strategies.

Step 7: Place Order

In order to place an order when the entry condition is met, we need to create a dictionary that contains the details for the order placement. This dictionary includes the following parameters:

```
request = {  
    "action": mt5.TRADE_ACTION_DEAL,  
    "symbol": 'EURUSD',  
    "volume": 0.2,  
    "type": mt5.ORDER_TYPE_BUY,  
    "price": mt5.symbol_info_tick('EURUSD').ask,  
    "comment": "Quantra Market Buy Order",  
}
```

- "action": This parameter specifies the type of trade action that you want to perform. In this case, we are using "mt5.TRADE_ACTION_DEAL", which means that the trade will be executed immediately at the current market price.
- "symbol": This parameter specifies the trading instrument that you want to trade. You need to replace "symbol" with the symbol name of the trading instrument you want to trade, such as "EURAUD" or "GBPUSD".
- "volume": This parameter specifies the lot size of the trade.
- "type": This parameter specifies the type of trade that you want to place, such as a buy/sell market order, limit order, or stop order.
- To place a market order, you can set "trade_type" to "mt5.ORDER_TYPE_BUY" or "mt5.ORDER_TYPE_SELL", depending on whether you want to buy or sell the trading instrument.

order. This can be useful for tracking your orders in the future.

Once you have set the parameters, you can use the "mt5.order_send()" function to send the trade request to the MetaTrader5 platform. The function will return a trade ticket number if the trade is executed successfully.

send the order request and check for errors

```
order_result = mt5.order_send(request)
```

```
if order_result.retcode != mt5.TRADE_RETCODE_DONE:
```

```
    print("Error placing order: ", order_result.comment)
```

```
else:
```

```
    print("Order placed successfully, order ticket:",  
order_result.order)
```

If you receive an error message that says "Automated trading is not enabled" when trying to place orders using the MetaTrader5 package in Python, you can follow these steps:

- Open the MetaTrader5 desktop application.
- Click on the "Tools" menu and select "Options."
- In the "Options" window, click on the "Expert Advisors" tab.
- Make sure that the "Allow algorithmic trading" checkbox is selected.
- Click "OK" to save the changes.

Once you've enabled automated trading, you should be able to place orders using the MetaTrader5 package in Python without any issues. You can also verify the execution of order from MetaTrader Terminal.

Symbol	Ticket	Time	Type	Volume	Price
eurusd	1681986161	2023.04.26 14:32:41	buy	1	1.10472
eurusd	1681986465	2023.04.26 14:32:48	buy	1	1.10476
eurusd	1681990586	2023.04.26 14:34:31	buy	0.2	1.10462
eurusd	1681991375	2023.04.26 14:34:51	buy	0.2	1.10477
eurusd	1681992007	2023.04.26 14:35:01	buy	0.2	1.10492
eurusd	1681994291	2023.04.26 14:35:39	buy	0.2	1.10476
eurusd	1681999371	2023.04.26 14:37:27	buy	0.2	1.10460
eurusd	1682001162	2023.04.26 14:38:17	buy	0.2	1.10476
eurusd	1682012298	2023.04.26 14:41:24	buy	0.2	1.10472

cookies. You can read more about our privacy policy here

```
result = mt5.positions_get()

if result:

    # create a list of dictionaries containing the data for each
    position

    data = pd.DataFrame([position._asdict() for position in result])

    print("Unrealized P&L: ", data.profit.sum())

    # print the DataFrame
    display(data.head())

else:

    print("No positions found")
```

This code retrieves the position information and calculates the unrealized P&L for all the open positions. If there are no positions found, the code will print a message indicating that no positions were found.

Step 9: Close Open Position

To close an open position, you can use the `order_send` function used previously. You also need to specify the ticket of the order or position you are closing. Here's an example code snippet that shows how to close an open position:

```
# connect to MetaTrader 5

mt5.initialize()

# get the ticket number of the position to close

ticket = int(data.iloc[0].ticket)

# check if the position exists and its type

position = mt5.positions_get(ticket=ticket)
```

cookies. You can read more about our privacy policy here

```
# if the position is a buy position, send a sell order to close it
```

```
request = {  
    "action": mt5.TRADE_ACTION_DEAL,  
    "symbol": position[0].symbol,  
    "volume": position[0].volume,  
    "type": mt5.ORDER_TYPE_SELL,  
    "position": position[0].ticket,  
    "price": mt5.symbol_info_tick(position[0].symbol).bid,  
}
```

```
else:
```

```
# if the position is a sell position, send a buy order to close it
```

```
request = {  
    "action": mt5.TRADE_ACTION_DEAL,  
    "symbol": position[0].symbol,  
    "volume": position[0].volume,  
    "type": mt5.ORDER_TYPE_BUY,  
    "position": position[0].ticket,  
    "price": mt5.symbol_info_tick(position[0].symbol).ask,  
}
```

```
# close the position
```

```
result = mt5.order_send(request)
```

```
if result.retcode != mt5.TRADE_RETCODE_DONE:
```

```
    print("Error closing position: ", result.comment)
```

```
else:
```

```
    print(f"Position {position[0].ticket} closed \
```

```
successfully, order ticket: {result.order}")
```

```
else:
```

```
    print(f"Position {ticket} not found")
```

If the position is closed successfully, the code will print a message indicating that the position was closed successfully. If the position close fails for some reason, the code will print an error message

reason for the failure.

cookies. You can read more about our privacy policy [here](#)

automated trading or specifically using MT5 Python integration, this guide will help you get started on your automated trading journey.

IMPORTANT DISCLAIMER: *This post is for educational purposes only and is not a solicitation or recommendation to buy or sell any securities. Investing in financial markets involves risks and you should seek the advice of a licensed financial advisor before making any investment decisions. Your investment decisions are solely your responsibility. The information provided is based on publicly available data and our own analysis, and we do not guarantee its accuracy or completeness. By no means is this communication sent as the licensed equity analysts or financial advisors and it should not be construed as professional advice or a recommendation to buy or sell any securities or any other kind of asset.*

CHECK OUT OUR FREE COURSES

cookies. You can read more about our privacy policy [here](#)

Learning for Trading

Co-authored by



Level: Beginner

Basic terminology, Research
Papers, Working Models

FREE

Options Trading Strategies In Python: Basic

Co-authored by



Level: Beginner

Covered Call, Protective Put,
Iron Condor, Bull Call

cookies. You can read more about our privacy policy here

Interactive Brokers Platform

Co-authored by



Level: Beginner

IBridgePy API, Installations,
Order & Portfolio Management

[About Us](#)

[QuantInsti](#)

[Blueshift](#)

[Associates](#)

[Contact Us](#)

[Affiliate Program](#)

[Privacy Policy](#)

[FAQs](#)

[Courses](#)

[Blog](#)

[For Business](#)

[Success Stories](#)

[Community](#)

[Glossary](#)

©2025 QuantInsti® - Quantra® is a
trademark property of QuantInsti®