

# MGMT 638

## Session 11

---

Kerry Back

Fall 2025

# Introduction to Alpaca-py

- Official Python SDK for Alpaca's API ecosystem
- Provides programmatic access to trading and market data
- Supports Python 3.8+
- Object-oriented design with built-in data validation using Pydantic
- Successor to `alpaca-trade-api-python` with improved architecture
- Free to use with free account signup at [alpaca.markets](https://alpaca.markets)
- Sandbox/paper trading environment available for testing

# Installation and Setup

## Installation:

- Install via pip: `pip install alpaca-py`
- Or via poetry: `poetry add alpaca-py`
- Current version: 0.43.2 (November 2025)

## Setup:

- Sign up for free account at [alpaca.markets](https://alpaca.markets)
- Retrieve API key and secret key
- Crypto market data requires no authentication
- Stock market data and trading require API keys

# Core Features and APIs

## Three Main API Products:

### 1. Market Data API

- Live and historical data for 5,000+ stocks, 20+ cryptocurrencies, and options
- Stock and crypto bar data, quotes, trades, and news

### 2. Trading API

- Execute trades in stocks, crypto, and options
- Fast execution speeds
- Order management and position tracking

### 3. Broker API

- Build investment applications
- Account creation and KYC verification
- Platform for robo-advisors or brokerage services

# Automated Trading Workflow 1: Scheduled Trading

## End-of-Day Trading Strategy:

1. **Download Data** (e.g., 30 minutes before market close)
  - Use Market Data API to fetch latest prices, quotes, and historical bars
  - Retrieve current positions and account information
2. **Generate Trading Signals**
  - Run local program (ML model, factor model, technical indicators)
  - Generate list of desired trades (buy/sell decisions and quantities)
3. **Execute Trades**
  - Use Trading API to submit market or limit orders
  - Batch order submission for multiple securities
  - Receive execution confirmations

# Automated Trading Workflow 2: Real-Time Monitoring

## Position Risk Management:

- **Stream Real-Time Data**

- Use WebSocket streaming clients for live price updates
- Monitor positions for extreme price movements
- Track market volatility and liquidity conditions

- **Detect Risk Events**

- Identify large losses or gains in positions
- Detect unusual volatility or gap moves
- Set threshold-based alerts

- **Automated Response**

- Submit stop-loss or take-profit orders automatically
- Rebalance positions to maintain target weights
- Send notifications for manual review

# Automated Trading Workflow 3: Continuous Trading

## High-Frequency or Algorithmic Strategies:

- **Continuous Data Stream**

- Real-time price feeds via WebSocket API
- Process tick-by-tick quotes and trades
- Minimal latency for execution speed

- **Continuous Signal Generation**

- Program runs perpetually during market hours
- Updates trading signals with each new data point
- May use mean reversion, momentum, or arbitrage strategies

- **Immediate Order Submission**

- Submit orders automatically when conditions are met
- Manage order lifecycle (cancel, replace, modify)
- Track fill rates and slippage

# Key API Features for Automated Trading

## Market Data API Features:

- Historical bars (minute, hour, day) for backtesting and signal generation
- Latest quotes and trades for current market conditions
- Real-time streaming via WebSocket for continuous monitoring
- News API for event-driven strategies

## Trading API Features:

- Order types: market, limit, stop-loss, bracket orders
- Batch order submission for portfolio rebalancing
- Position and account queries for portfolio management
- Order status tracking and cancellation
- Paper trading environment for testing strategies



# Key Modules and Resources

## Main Modules:

- `alpaca.trading` – Order submission, position management
- `alpaca.data.historical` – Historical bar data, quotes, news
- `alpaca.data.requests` – Request object definitions
- `alpaca.broker` – Account creation, KYC verification

## Documentation:

- [Official alpaca-py Documentation](#)
- [GitHub: alpaca-py Repository](#)
- [Alpaca API Documentation](#)

# Example Jupyter Notebooks

## Official alpaca-py Examples:

- [All Examples Directory](#) – Stocks, crypto, options
- [Stock Trading Basics](#) – Notebook for stock trading
- [Crypto Trading Basics](#) – Notebook for crypto trading
- [Options Trading Basics](#) – Notebook for options trading

## Additional Strategy Examples:

- [Alpaca Notebooks Repository](#) – Crypto arbitrage, pairs trading, momentum strategies, and more