ENVERUS

USER GUIDE

MARKETVIEW ADVANCED DESKTOP PLUGIN USER GUIDE

Company Overview

ENVERUS

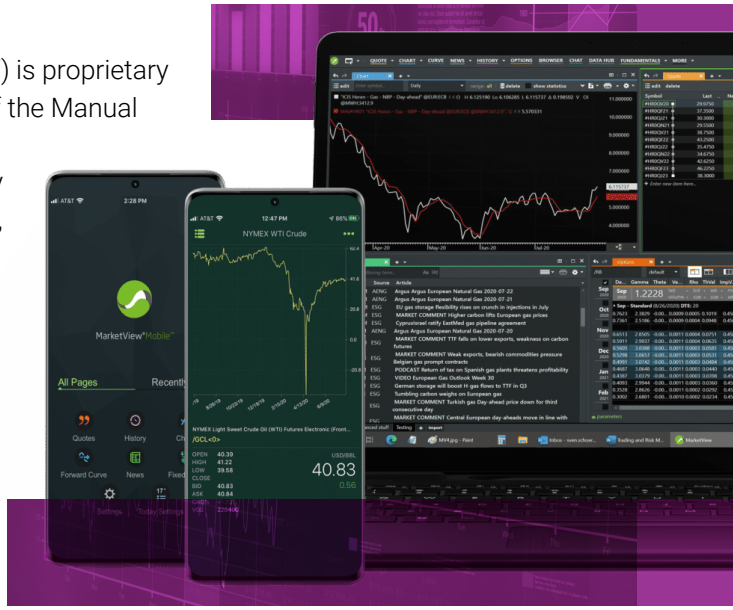
MarketView by Enverus is an award-winning provider of desktop analytics, risk management, and data integration solutions for the global commodity trade. Some of the largest energy and commodity companies in the world trust MarketView by Enverus to manage millions of data points, both public and customer proprietary, in support of risk management and trading decision support on an enterprise-wide basis. MarketView by Enverus offers an extensive library of energy andv commodity data sets including real-time exchange, price reporting agencies, brokers, government reporting, emerging markets, and specialty data.

Founded in 1996, MarketView by Enverus has been an industry leader in energy and commodity data management solutions that fully integrate real-time and historical market data and customer proprietary data using the internet and enterprise network systems. We focus exclusively on the energy and agricultural industries, and pride ourselves in understanding how our customers use and analyze commodity market information to operate their businesses. With offices in Chicago, Houston, Calgary, Sao Paulo, London, and Singapore, MarketView by Enverus is advantageously positioned to support its customers globally.

USER MANUAL DISCLAIMER

The MarketView Advanced Desktop Plugin User Manual (the "Manual") is proprietary to Enverus and no ownership rights are hereby transferred. No part of the Manual shall be used, reproduced, translated, converted, adapted, stored in a retrieval system, communicated or transmitted by any means, for any commercial purpose, including without limitation, sale, resale, license, rental or lease, without the prior express written consent of Enverus.

MarketView by Enverus does not make any representations, warranties or guarantees, express or implied, as to the accuracy or completeness of the Manual. Users must be aware that updates and amendments will be made from time to time to the Manual and it is the user's responsibility to determine whether there have been any such updates or amendments. Enverus reserves the right to make changes to any and all parts of this publication at any time, without any obligation to notify any person or entity of such changes.

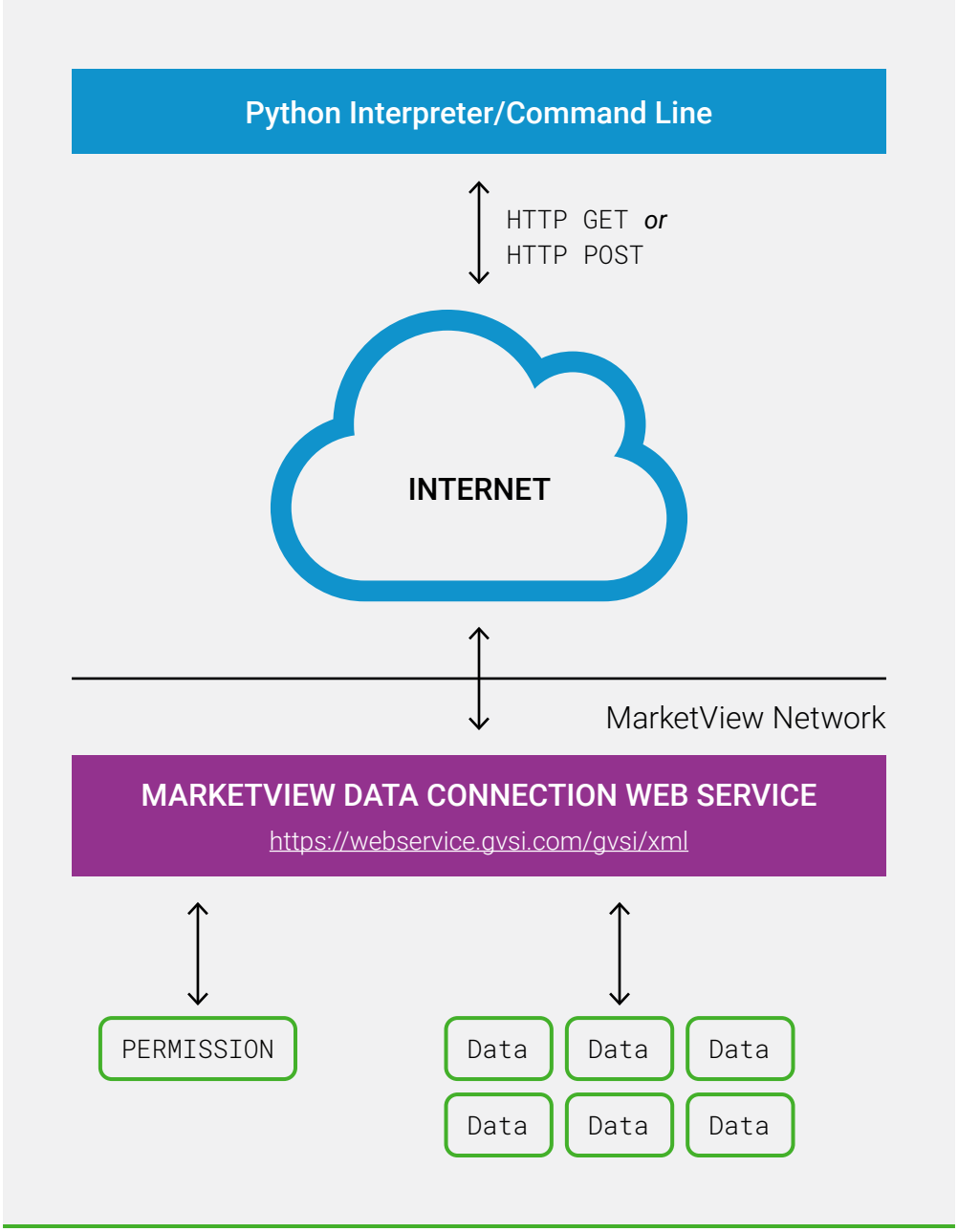


Overview and Architecture

PYTHON OVERVIEW

The MarketView Advanced Desktop Plugin allows customers to leverage the power of Python combined with a robust and nimble Web Service to pull in and perform in-depth, complex analysis on Futures, Spot, Forward Curve, and Futures Options prices from a comprehensive selection of MV data.

DATA FLOW DIAGRAM



Requirements

MINIMUM REQUIREMENTS

- MarketView Advanced Desktop Plugin is solely compatible with Python 3 or higher.
- Open-source library ***"Requests"***. If the user does not have this installed, open the command prompt and type, ***"pip3 install requests"***. If you do not have ***"Requests"*** downloaded and saved locally, please download here: https://github.com/requests/requests.
- MarketView will set the user up with a login and password, as well as permission the component that allows them to have access to the **MarketView Web Service API**, which is required for the Python SDK.

SDK Files and Classes

GvWSConnection.py: This file contains the main class for communication with the Web Service called GvWSConnection.

- **GviResult** class – This is a helper class that represents one row in a returned data set.
- **ConvertedSymbol** class – This class helps to define all the unit and currency conversions for a given symbol.
- **Enumerator** classes – These help users define particular parameters that must be used when requesting certain types of data.
    - » "QuoteFields" lists all the fields permitted in Quote requests.
    - » "CurrencySources" will list all the supported unit and currency options for a given symbol.
    - » "TimeSeriesFields" will list the supported fields for Daily, Weekly, Monthly, Quarterly, etc. aggregations.

gwsExample.py: This file contains examples of each type of request that is supported by the Python SDK. It allows for a basic understanding of symbols, fields, unit/currency conversions, etc. that a user can request

Error Codes

ERROR CODE	ERROR TYPE
101	Invalid credentials
102	Invalid request format
401	User unauthorized
405	HTTP method is not allowed
408	Request Timeout

Authentication

```
import datetime
from GvWSConnection import QuoteFields, GvWSConnection,
ConvertedSymbol, Currencies, CurrencySources, Units

conn = GvWSConnection("yourusername", "yourpassword")

USERNAME = 'yourusername'
PASSWORD = 'yourpassword'
```

Outputting Data

As shown in the ***'gws_example.py'*** file, you can build a table and send data to it with the prepackaged code we have below:

```
def pretty_print_table(table):
    items_len = len(table[0])
    lengths = [0] * items_len
    for item in table:
        for index in range(items_len):
        l = len(item[index])
        if lengths[index] < l:
            lengths[index] = l

    format_str = ""
    for ln in lengths:
        format_str += "{:>" + str(ln + 1) + "}|"

    header = format_str.format(*table[0])
    separator = '-' * len(header)
    print(separator)
    print(header)
    print(separator)
    for row in table[1:]:
        row_str = format_str.format(*row)
        print(row_str)
    print(separator)
        print("")

def print_results(result, start_column=0):
    table = [result[0].field_names[start_column:]]
    for row in result:
        values = list(row.values())
        str_values = [str(x) for x in values[start_column:]]
        table.append(str_values)
pretty_print_table(table)
```

Python Basic Request Types

GetQuote

This request will return the latest pricing for a given symbol for all available requested fields.

```
conn = GvWSConnection("yourusername", " yourpassword ")
quote_result = conn.get_quote('/GCL')
q = quote_result[0]
print("symbol: {} open: {} high: {} low: {} last: {} date: {}".format(
    q.symbol, q.open, q.high, q.low, q.last, q.trade_date))

print("Done!")
```

GetQuote Currency Conversion

This request will return the latest pricing for a given symbol for all requested fields available. In addition, it converts the currency and lot units to US Cents per Gallon instead of the default Currency and Lot Unit.

```
print("Latest Data for Crude (Converted), Heating, Oil, and Nat Gas")
gcl_converted = ConvertedSymbol("/GCL", currency=Currencies.USC,
currency_
source=CurrencySources.USF, unit=Units.GAL)

q2 = conn.get_quote([gcl_converted, '/GHO', '/GNG'])
print_results(q2)
```

GetDaily

This request will return the last ten days of data for the symbol(s) of your choice

```
conn = GvWSConnection("yourusername", " yourpassword ")
print("Last 10 Days of NYMEX Benchmark Data:")
d1 = conn.get_daily(['/GCL','/GRB','/GNG'], num_of_bars=10)
print_results(d1)
```

GetDaily Date Range

This request will return the range of data you designate for the symbol(s) of your choice.

```
conn = GvWSConnection("yourusername", " yourpassword ")
from_date = datetime.date(2021, 9, 1)
to_date = datetime.date(2021,9, 8)

d2 = conn.get_daily_range('/GCL', from_date, to_date)
print_results(d2)
```

GetForwardCurve

This request will return the entire forward curve for the symbol root you pass. You are required to pass the curve date, which will display the entire curve "as of" an effective date.

```
conn = GvWSConnection(USERNAME, PASSWORD)
curve_date = datetime.date(2021, 9, 5)
print("Forward curve for /GCL, date: %s" % (curve_date.strftime("%Y-%m-
%d")))
f1 = conn.get_forward_curve('/GCL',curve_date)
print_results(f1)
```

GetChain

Please be sure to have QuoteFields and TimeSeriesFields Classes defined

This request will return a snapshot (most recent pricing) for an entire curve.

```
conn = GvWSConnection(USERNAME, PASSWORD)
print("Snapshot of /GCL Data")
f1 = conn.get_curve('/GCL',fields=QuoteFields.STANDARD)
print_results(f1)
```

GetIntraday

This request will return hourly time series data for /GCL for the last five days. Time intervals can be requested as 5, 15, 30, 60 minute bars.

```
conn = GvWSConnection(USERNAME, PASSWORD)
print("Crude Oil Hourly Bars")
d2 = conn.get_intraday('/GCL', bar_interval=60,days_back=5)
print_results(d2)
```

GetIntraday Date Range

This request will return hourly time series data for /GCL for the date range of your choice. Time intervals can be requested as 5, 15, 30, 60 minute bars

```
from_date = datetime.date(2021, 9, 1)
to_date = datetime.date(2021,9, 8)
d2 = conn.get_intraday('/GCL', bar_interval=60,start_date=from_date,end_
date=to_date)
print_results(d2)
```

Enverus delivers business-critical insights to the energy, power, and commodities markets. Its state-of-the-art SaaS platform offers sophisticated technology, powerful analytics, and industry-leading data. Enverus's solutions deliver value across upstream, midstream and downstream markets, empowering exploration and production (E&P), oilfield services, midstream, utilities, trading and risk, and capital markets companies to be more collaborative, efficient, and competitive. Enverus delivers actionable intelligence over mobile, web, and desktop to analyze and reduce risk, conduct competitive benchmarking, and uncover market insights. Enverus serves over 5,000 companies globally from its Austin, Texas, headquarters and has more than 1,000 employees. For more information visit enverus.com.