T STOCK V2.0

Tech Note

Ming

2018

Content

History

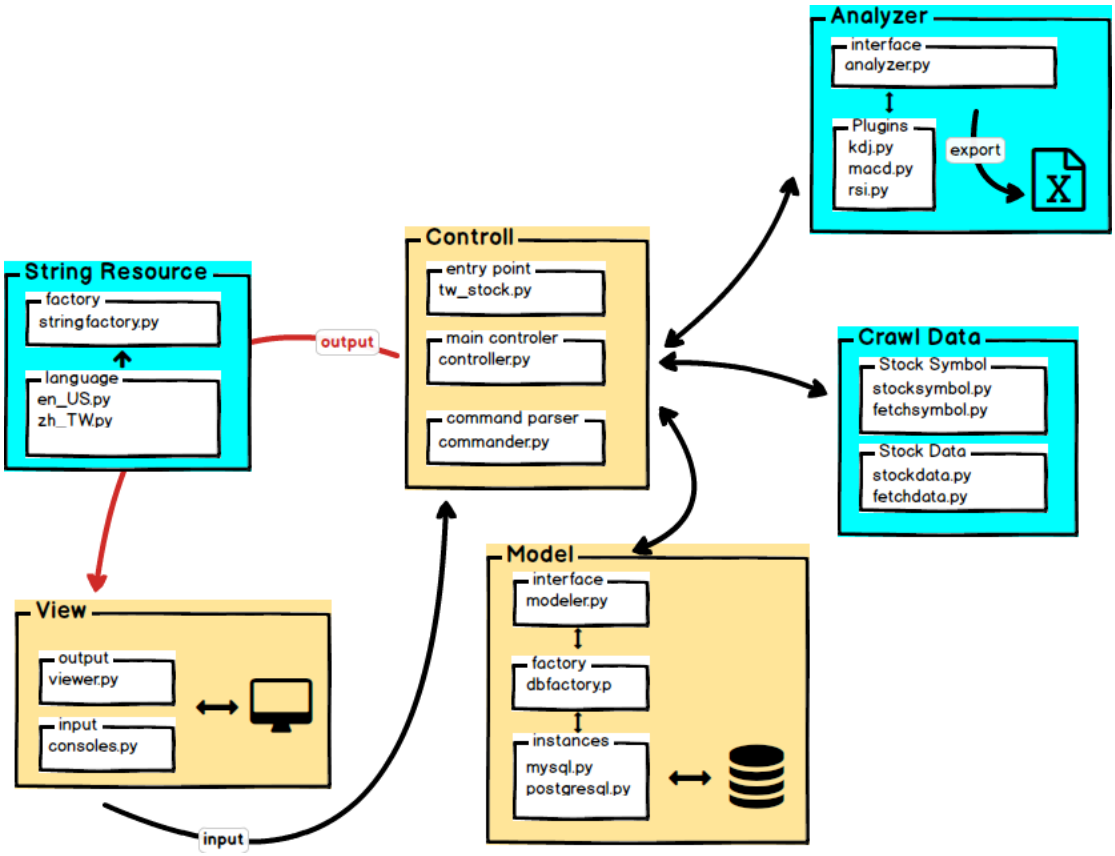
Revision	Common	Date
V1.0	Initial release	2018/8/26

1. Abstract

This document will describe the SW architecture, design concept and how to extend the locale language, RMDB interface & stock technical analysis.

2. SW architecture

In the startup stage, my goal is using MVC concept to develop, but misunderstand that and become to below morph concept, I should move the “Analyzer” & “Crawl Data” components to “MODEL” layer.

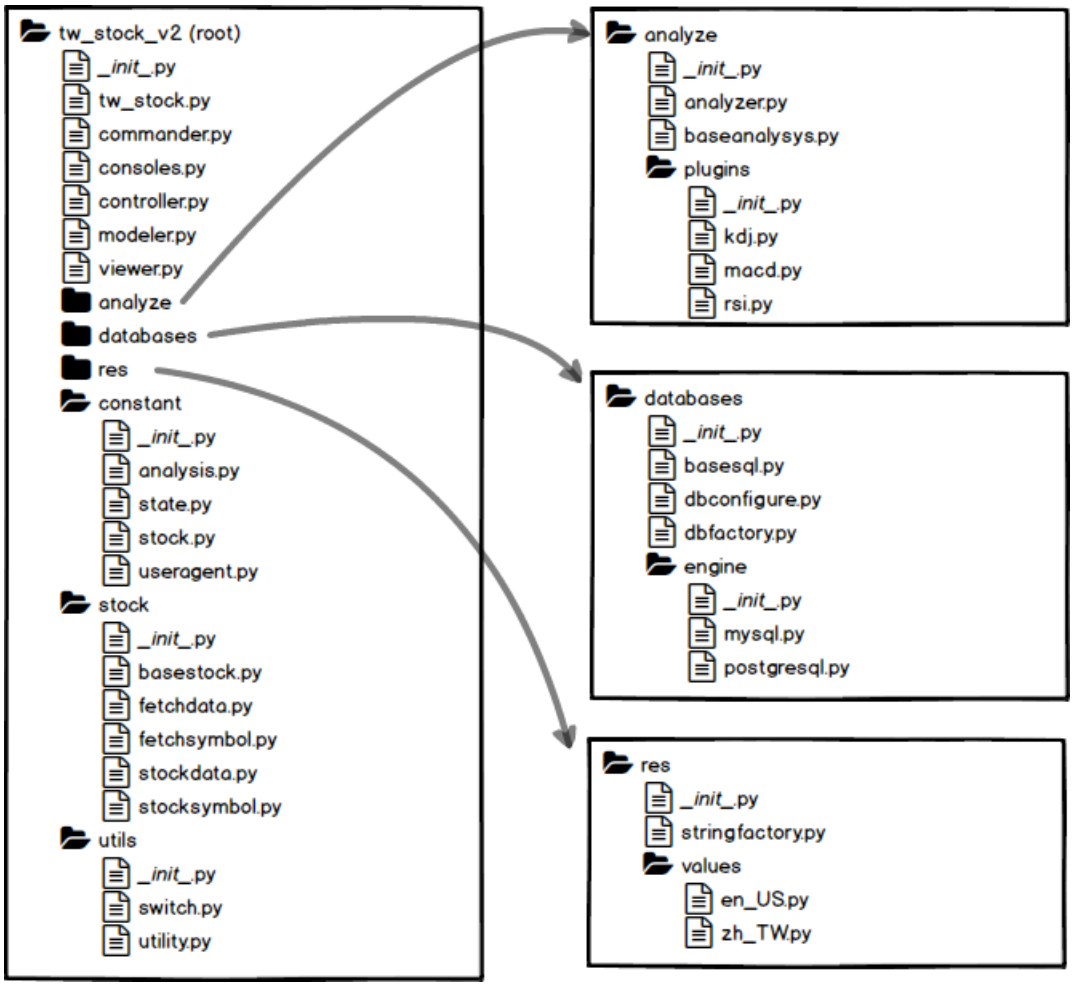


3. Learned in this project?

- The “xpath” method to sort the crawl data from web site.
- The “colorama” package for “ANSI” escape sequences.
- The “DataFrame” type of “pandas” package.
- Multi-processing & queue packages to parallel crawl data, calculate stock data.
- Dynamic loading the module and package.
- Connecting to RMDB (MySQL, PostgreSQL) by Python DB API.

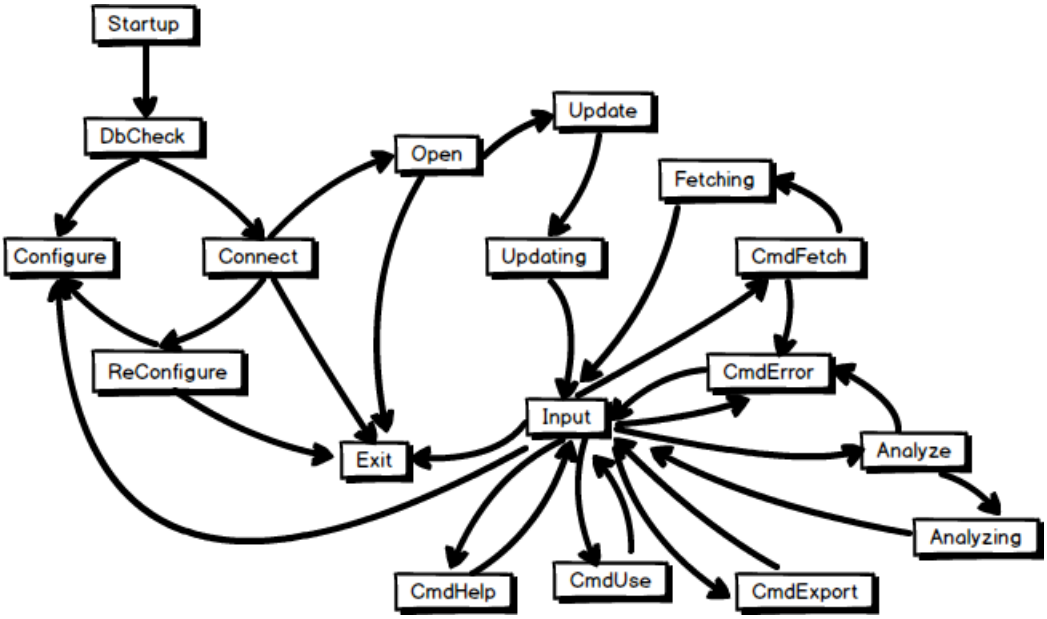
- PEP 8 – Style guide for Python code
- Crawl web data by “urllib” package.
- Extract data by JSON format.
- Inherit (single), abstract method & property.
- Read/write XML file by “lxml” package.
- Replacements for switch statement in Python

4. File structure



5. State Machine

I using the “state machine” to control the flow, after review that by self, it should have many parts can be improving.



6. How to extend the locale resource

In this project, I used “dictionary” type to store the locale resource, you can found “en_US.py” and “zh_TW.py” under “res/values” path, the “en_US” & “zh_TW” are the focus for extend the locale resource.

If you didn’t understand which “locale” was configured at your operation system, you can follow below steps to check the “locale” setting in your computer, from the below example, the “**zh_TW**” is my computer locale configuration.

```
C:\Users\Ming>python
Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2018, 17:00:18) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import locale
>>> print (locale.getdefaultlocale())
('zh_TW', 'cp950')
>>>
```

For example, if you want to add German resource, you just need to copy the “en_US.py” file and rename to “de_DE.py”, then translate the English contents to German.

In the locale resource file, you may found some specific symbols, such as “%d” or “%s”, please be careful to do the translation.

Key	zh_TW	en_US	*de_DE*
STR_DB_CONNECTED	%s 資料庫已連線	%s database connected	***%s datenbank verbunden***

7. Database Structure

In this chapter, will describe the schema of the database, the database name is “tw_stock”, the stock symbols table is call the “stock_symbol”.

For the stock data, the table name is using “symbol_[stock symbol]” as the format, for example, “symbol_2330” table will store all “TSMC” bargain records, and the “symbol_2498” table will be store all “HTC” bargain records.

7.1 Schema of the “stock_symbol”

Column Name	#	Data Type	Not Null	Auto Increment	Key	Default	Extra
id	1	int(10) unsigned	☑	☑	PRI		auto_increment
symbol	2	varchar(10)	☐	☐	UNI	NULL	
name	3	varchar(32)	☐	☐		NULL	
create_date	4	varchar(10)	☐	☐		NULL	
update_date	5	varchar(10)	☐	☐		NULL	

7.2 Schema of the “symbol_[xxxx]”

The “[xxxx]” mean the stock symbol, such the table of TSMC as “symbol_2330”.

Column Name	#	Data Type	Not Null	Auto Increment	Key	Default	Extra
id	1	int(10) unsigned	☑	☑	PRI		auto_increment
trade_date	2	varchar(10)	☐	☐	UNI	NULL	
trade_volumn	3	varchar(16)	☐	☐		NULL	
trade_money	4	varchar(16)	☐	☐		NULL	
trade_open	5	varchar(8)	☐	☐		NULL	
trade_max	6	varchar(8)	☐	☐		NULL	
trade_min	7	varchar(8)	☐	☐		NULL	
trade_end	8	varchar(8)	☐	☐		NULL	
trade_spread	9	varchar(8)	☐	☐		NULL	
trade_count	10	varchar(10)	☐	☐		NULL	

8. How to support the 3rd RMDB

In this chapter, will describe how to extend the interface for other RMDB service, please refer the “mysql.py” and “postgresql.py” files, and the 3rd RMDB service must meet below requirement for TW_STOCK_V2.

- Relate python package supported
- SQL command supported
- PEP-249 (Python Database API Specification v2.0) supported

8.1 Modify the SQL statement

In this step, you should need to re-write some SQL commands about check/create DATABASE and TABLES, you can refer the “mysql.py” and “postgresql.py” files, we can found a little difference on SQL statements.

- CMD_CHECK_DATABASE
- CMD_CREATE_DATABASE
- CMD_CHECK_TABLE
- CMD_CREATE_SYMBOL_TABLE
- CMD_CREATE_DATA_TABLE

8.2 Modify the name() function

For this part, just let user know which RMDB service will be used.

MariaDB/MySQL	def name(self): return **"MariaDB/MySQL"**
PostgreSQL	def name(self): return **"PostgreSQL "**

8.3 Modify the dependency() function

About this part, let TW_STOCK_V2 can recognize which package for this RMDB, and you can provide multiple name, if the 1st package not installed, will follow the 2nd package.

MariaDB/MySQL	def dependency(self): return ['MySQLdb', 'pymysql']
PostgreSQL	def dependency(self): return ['psycopg2']

8.4 Modify “connect” & “open” functions

Let us describe the “connect()” & “open()” functions first, we may need case by case to modify “connect()” & “open()” function for each RMDB system.

- ***connect()***:In first time, the “tw_stock” database is not exist, so we can’t connect to RMDB and using “tw_stock” by default, we need to check the “tw_stock” and created it.
- ***open()***: Assign the “tw_stock” database for future operation.

	MariaDB/MySQL	PostgreSQL
Connect	DB name is not mandatory	DB name is mandatory
Open	Use “select_db()” function to select DB	Reconnect with new DB name

From above table, we need to clarify how to create database/table and select new DB for operation.

9. How to extend the “Tech. Analysis”

We will explain how to add new “Technical Analysis” items into TW_STOCK_V2, we suggest you can refer “kdj.py”, “macd.py” & “rsi.py” from “analyze/plugins” folder, I will using “rsi.py” for example.

9.1 Construct function

In this part, we just need to modify the “rsi” statement to new class name.

```
class rsi(Process, BaseAnalyer):
    def __init__(self, arg_share_data, arg_queue, **kwargs):
        super(rsi, self).__init__()
        self.__data = arg_share_data
        self.queue = arg_queue
    # The “**kwargs” feature was not implemented.
```

9.2 The name of technical analysis

Just need to modify the technical analysis name for “analysis ?” query command.

```
def analysis_name(self):
    return "RSI"
```

9.3 The columns name of technical analysis result

Provide the columns name of analysis result, we need to use this info to assign the columns name to “DataFrame” type.

```
def colnum_info(self):
    return ("RSI",)
```

9.4 Calculate the technical analysis data

We must use “DataFrame” type to response the calculated result.

```
def run(self):
    super(rsi, self).delay()    # To avoid the crash on multi-processer thread.
    self.__df_result = DataFrame([0] * len(self.__data))    # Allocated the result size
    ## Add your Calculate on this section
    self.__df_result.columns = self.colnum_info()    # Assign the column name
    self.queue.put([RetriveType.DATA, self.__df_result])    # Use “Queue” to pass result
    self.queue.put([RetriveType.INFO, [self.analysis_name(), Info.INFO_CALCULATED]])
```

About the “self.__data”, it’s a “list” type, you can using “*self.__data[row][column]*” statement to get the value.

id	trade_date	trade_volumn	trade_money	trade_open	trade_max	trade_min	trade_end	trade_spread	trade_count
1	2018/04/16	3,680,369	86,460,520	23.75	24.20	22.60	23.90	X0.00	2,198
2	2018/04/17	1,299,890	29,840,564	23.90	23.90	22.45	22.55	-1.35	826
3	2018/04/18	887,856	19,236,982	22.80	22.80	20.80	21.60	-0.95	562
4	2018/04/19	354,000	7,556,450	21.05	21.60	21.05	21.30	-0.30	244
5	2018/04/20	276,709	5,911,513	21.30	21.75	20.90	21.15	-0.15	207
6	2018/04/23	346,497	7,222,210	21.50	21.55	20.60	20.60	-0.55	223
7	2018/04/24	536,875	10,730,337	20.80	20.80	19.70	19.85	-0.75	277
8	2018/04/25	301,100	6,001,710	19.95	20.10	19.65	20.10	+0.25	151
9	2018/04/26	214,277	4,282,233	20.25	20.50	19.85	19.90	-0.20	115
10	2018/04/27	130,019	2,587,330	19.95	20.00	19.80	19.95	+0.05	77
11	2018/04/30	75,149	1,512,009	19.95	20.25	19.95	20.25	+0.30	50
12	2018/05/02	67,000	1,370,000	20.25	20.70	20.25	20.35	+0.10	46
13	2018/05/03	114,210	2,299,889	20.35	20.40	20.00	20.05	-0.30	64
14	2018/05/04	30,000	603,200	20.05	20.20	20.05	20.10	+0.05	26