WorkFlow Training II
Market Configurations

Leo Liu
lliu@cmcrc.com

Tools you need

1. TRTH (tick by tick data provided by TR)
2. Python3.6
3. Git (version control system)
4. Jira (communication system)
5. Bitbucket (code repository)
6. Pycharm (python IDE)
7. Jupyter notebook (optional)

Bitbucket student account:

cmcrc-student

Cmcrctraining!

What and Why?

This series of training is to prepare you to do research and work with your industry sponsor. MQD is a one-stop research platform. So this teaches you in depth how to use MQD to help you with your research.

MQD streamlines the process of research, from data download, data processing, producing results to the presentation of research.

Any data needs cleaning and filtering before it can be used to produce results.

For market microstructure, we need to clean up the data from the exchange. When we say “setting up a market”, it essentially means setting up a correct way to clean and filter the data for a particular exchange in MQD.

WorkFlow Engine

Luckily, Central Dev Team invented workflow engine to automate the data downloading and processing. You will just need to define the parameters in the system configuration for the workflow to do the hard work for you.

The workflow of workflow engine (A recap):

1. Download data (which asset class? Which securities you want to download?)
2. Convert data to blueshift (an efficient data format for storing and reading)
3. Generate results and store in S3 (data storage service by AWS).
4. Sync results to MQD backend database.
5. Display on MQD-QA website.

The process of setting up the market

1. **Build up the correct market configuration files - yaml file (this session)**
2. Create dependencies , tasks and jobs (session 3)
3. Run testing month
4. Monitor WF jobs and debugging errors. (session 4)
5. QA (Quality Assurance) the results. (session 5 and 6)

Market Configurations - setting up yaml file

What are the parameters you need to tell the WorkFlow Engine?

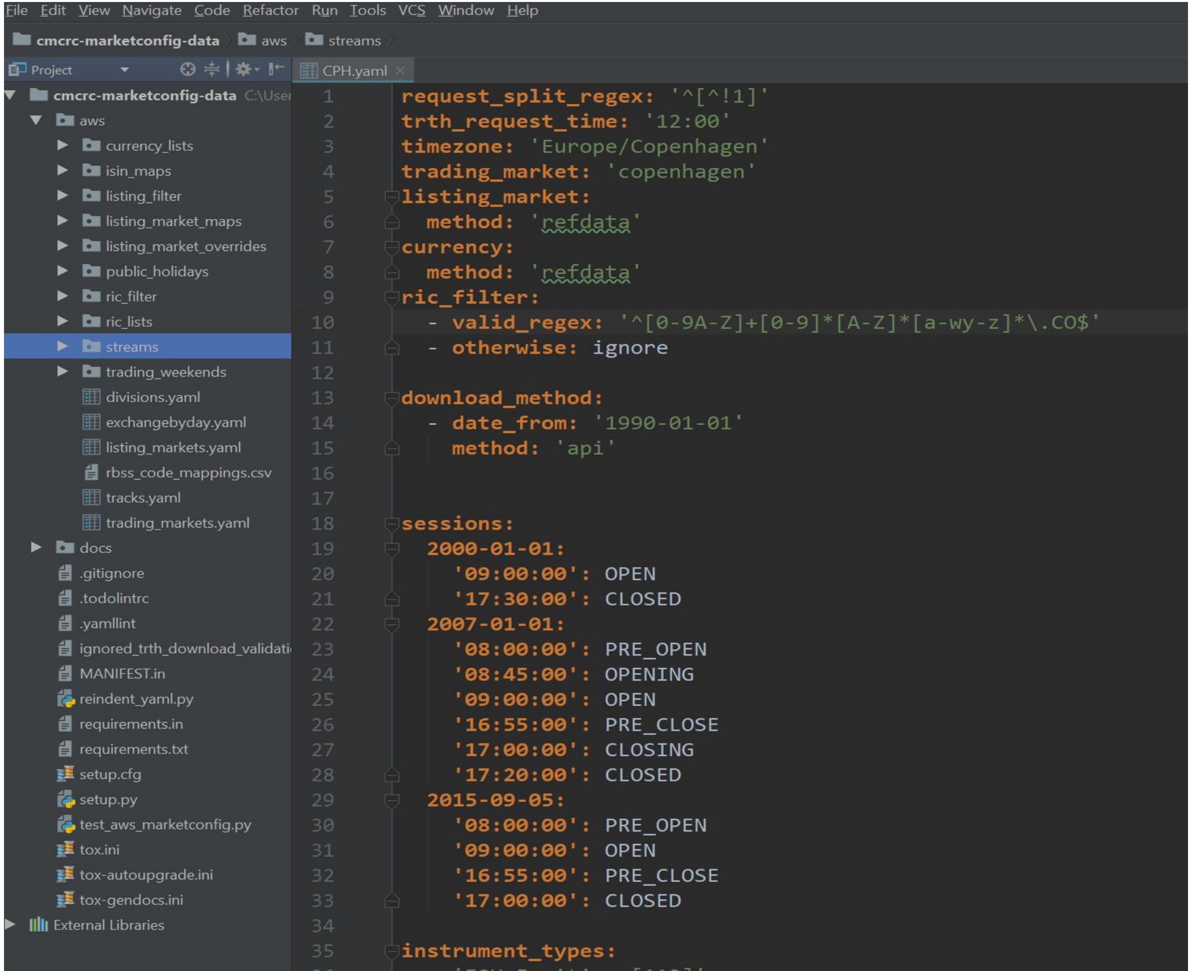
1. Trading hours
2. Securities you want to include
3. Transaction types (on or off, dark trade?)
4. and … many more

We will go through them one by one.

The structure of yaml files

1. Stream.yaml - To define the parameters in the data feed of TRTH
2. Trading_market.yaml - Defines a trading market
3. Listing_market.yaml - Defines a listing market
4. Tracks.yaml - Define how streams should form a trading market.

In most cases, a trading market is formed by one stream. For example, ASX trading market is formed by ASX in TRTH data feed. It is also quite often that several streams one trading market.


File Edit View Navigate Code Refactor Run Tools VCS Window Help
cmcrc-marketconfig-data
aws
streams
Project
CPH.yaml

```
cmcrc-marketconfig-data C:\User
  aws
    currency_lists
    isin_maps
    listing_filter
    listing_market_maps
    listing_market_overrides
    public_holidays
    ric_filter
    ric_lists
    streams
    trading_weekends
    divisions.yaml
    exchangebyday.yaml
    listing_markets.yaml
    rbss_code_mappings.csv
    tracks.yaml
    trading_markets.yaml
  docs
  .gitignore
  .todolintrc
  .yamllint
  ignored_trth_download_validati
  MANIFEST.in
  reindent_yaml.py
  requirements.in
  requirements.txt
  setup.cfg
  setup.py
  test_aws_marketconfig.py
  tox.ini
  tox-autoupgrade.ini
  tox-gendocs.ini
External Libraries
```

```yaml
request_split_regex: '^[^!1]'
trth_request_time: '12:00'
timezone: 'Europe/Copenhagen'
trading_market: 'copenhagen'
listing_market:
  method: 'refdata'
currency:
  method: 'refdata'
ric_filter:
  - valid_regex: '^[0-9A-Z]+[0-9]*[A-Z]*[a-wy-z]*\.CO$'
  - otherwise: ignore

download_method:
  - date_from: '1990-01-01'
    method: 'api'


sessions:
  2000-01-01:
    '09:00:00': OPEN
    '17:30:00': CLOSED
  2007-01-01:
    '08:00:00': PRE_OPEN
    '08:45:00': OPENING
    '09:00:00': OPEN
    '16:55:00': PRE_CLOSE
    '17:00:00': CLOSING
    '17:20:00': CLOSED
  2015-09-05:
    '08:00:00': PRE_OPEN
    '09:00:00': OPEN
    '16:55:00': PRE_CLOSE
    '17:00:00': CLOSED

instrument_types:
```

Stream yaml file

The parameters are stored in what we call “yaml file”. There are several of them, we will look at each of them.

First, the stream yaml file (market specific configuration file). The name of these files is corresponding to the feed name in TRTH, normally, they are the name abbreviation for the exchange for example ASX.yaml.

Those are the parameters defined in this file

1. Trth_request_time - the time of workflow engine to request data each day
2. Request_split_regex - filter in the download step

Stream yaml file

3. Timezone - The time zone info for the exchange

4. Trading_market - trading market information

5. Listing_market - listing market information

6. Currency - currency information for each security.

7. Download_method - Downloading using api or exchange by day

8. Instrument_types - Instrument types to include in the download

Stream yaml file

9. RIC_filter - Further filtering during convert step.

Note RIC stands for Reuters Instrument Code

10. Pre_conversion_rules - Some rules to do certain data corrections

11. Post_conversion_rules - Some rules to do certain data corrections

12. Sessions- Trading hours

13. Qualifiers - label attached to each transaction to signal the transaction type

Stream yaml file

1. Trth_request_time

The time of workflow engine to request data.

It is normally the midnight of the day. For most market, leave it as it is in the current stream yaml.

Consult BA team if you do find this causes problems.

Time zone

https://en.wikipedia.org/wiki/List_of_tz_database_time_zones

This list contains all timezones, simply find your market location there.

Filtering

2. request _split_regex and RIC_filter

request_split_regex defines the regular expression to do RIC (reuters identification code) filtering when downloading from TRTH.

RIC_filter defines the procedure to do RIC filtering when converting the data.

(Why is it better to filter in convert step rather than during download?)

Suppose you only want equity data, let's say equity stocks and ETFs, how do you exclude all other stuff ?

Noted that the two filters are in Download and Convert step. Additional filtering can happen in Sync step which is defined in list/trading_market.yaml file.

Filtering

First, you would need to know what securities you need to download e.g. valid security list (get such information from exchange website)

You will need to try to extract this list of securities from TRTH by writing a regular expression - a string expression to match multiple strings.

(why don’t we use a fixed list)

<< brief regex demonstration >>

Additional Resource: https://regex101.com/

Filtering

OK, let’s get our hands dirty and using Copenhagen as a real example.

1. Suppose we want to only include equities (Common stocks and ETFs). What you want to download? Let’s look at the official list of equity from the exchange.
2. Feed the list of securities to TRTH to get a feel of what do they look like, so we can figure out the right regex.

<< Demonstrate how to import security list to TRTH>>

Filtering

Based on the observation, let’s figure out the regex.

<< Demonstrate writing the regex for Copenhagen >>

This is a small market and not many RICs. we would prefer downloading everything except for obvious junk RICs. In this case, anything start with ! or 1.

Therefore, we will apply our RIC filter in the convert step.

RIC filter rule book

Basic rules:

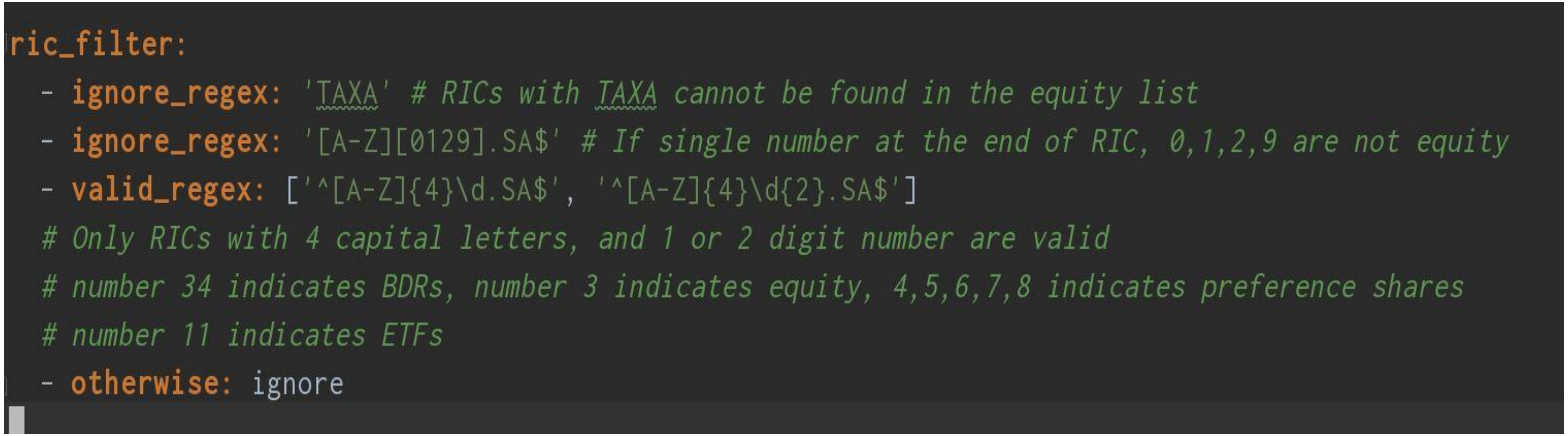
ignore — The security or securities listed are ignored.
valid — The security or securities listed are accepted.
otherwise — Either the word “valid” or “ignore”, setting the default for all other listings. This must be the last rule.

RIC filter rule book

Modifiers:
csv — Read a list from the CSV file given.
regex — Interpret the value as a regular expression.
not — Invert the meaning of the test. The securities will be ignored or valid if they are not listed, or do not match the regex.
condition — Only apply the rule if the condition is met. Conditions currently support date_from, date_to, and instrument_type

As far as the “otherwise” rule is concerned:
It is guessed if all the other rules are of the same kind (ie, all “ignore” or all “valid”).
If there is a mix of “ignore” and “valid” rules, there must be an “otherwise” rule.

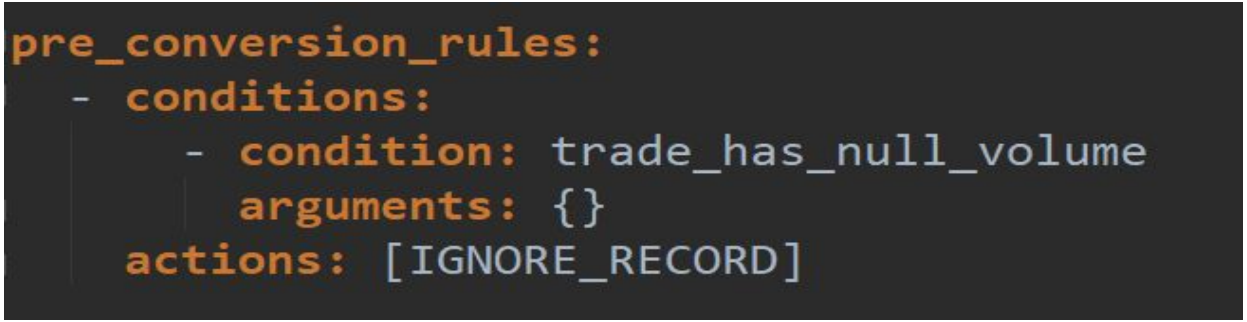
RIC filter



```
ric_filter:
  - ignore_regex: 'TAXA' # RICs with TAXA cannot be found in the equity list
  - ignore_regex: '[A-Z][0129].SA$' # If single number at the end of RIC, 0,1,2,9 are not equity
  - valid_regex: ['^[A-Z]{4}\d.SA$', '^[A-Z]{4}\d{2}.SA$']
  # Only RICs with 4 capital letters, and 1 or 2 digit number are valid
  # number 34 indicates BDRs, number 3 indicates equity, 4,5,6,7,8 indicates preference shares
  # number 11 indicates ETFs
  - otherwise: ignore
```

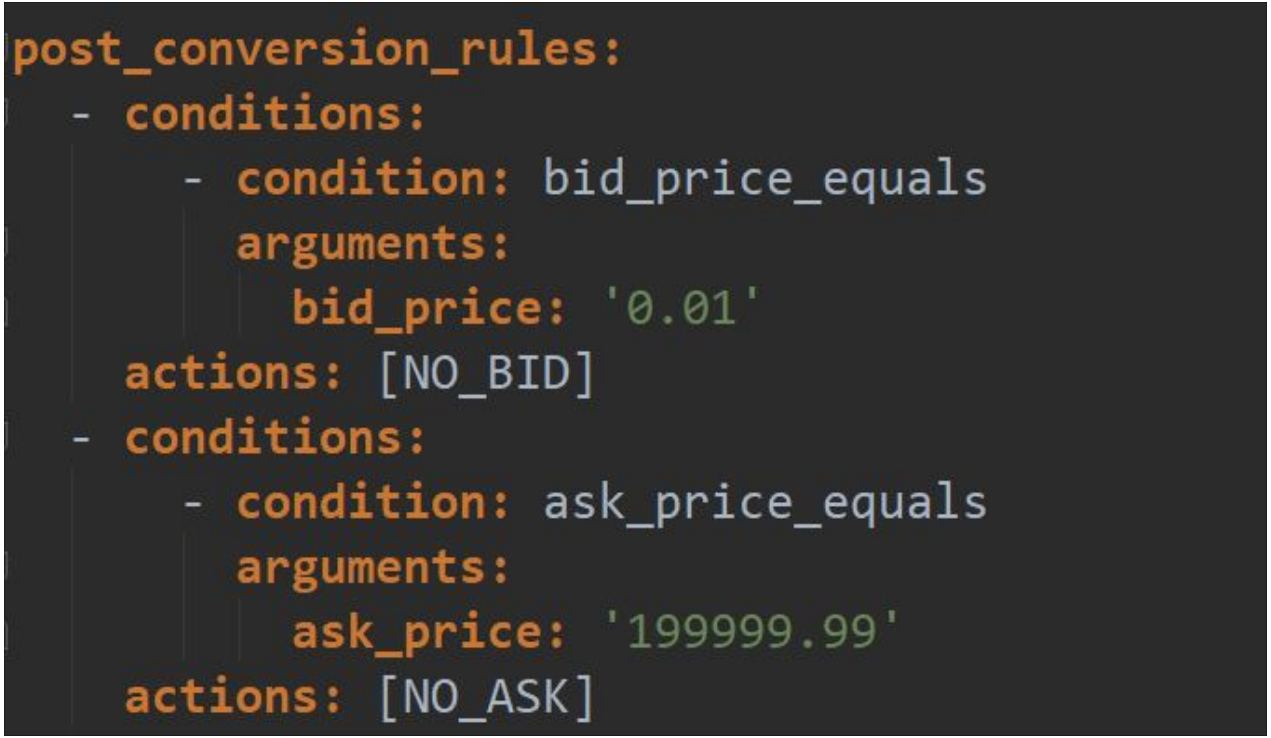
Pre or post-conversion rules

These rules are created to do data corrections. Consult BA team when you want to do certain corrections.



```
pre_conversion_rules:
  - conditions:
      - condition: trade_has_null_volume
        arguments: {}
    actions: [IGNORE_RECORD]
```

Pre or post-conversion rules



```yaml
post_conversion_rules:
  - conditions:
      - condition: bid_price_equals
        arguments:
          bid_price: '0.01'
    actions: [NO_BID]
  - conditions:
      - condition: ask_price_equals
        arguments:
          ask_price: '199999.99'
    actions: [NO_ASK]
```

Download method

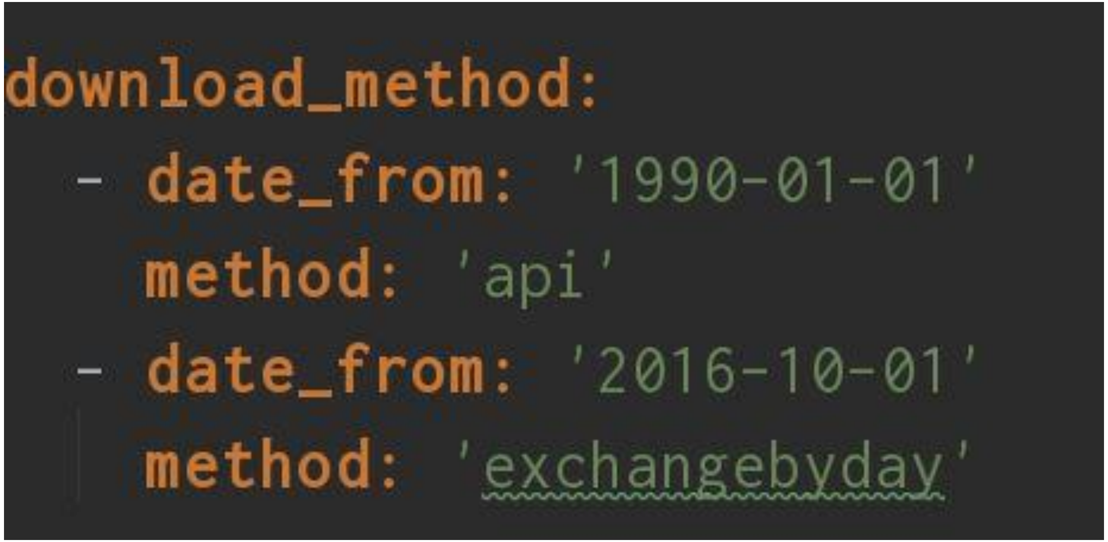
We have two methods of downloading data from TRTH in WorkFlow: API or exchangebyday.

API download will download each market separately. You will need create jobs for each market separately.

Exchangebyday will download all market in one go, and leave to convert job to separate them into each market.

For most markets, we have switch from API download to Exchangebyday in Oct 2016. It includes TAQ, depth, end-of-day, corpaction data. We still using API for download index data.

Download method



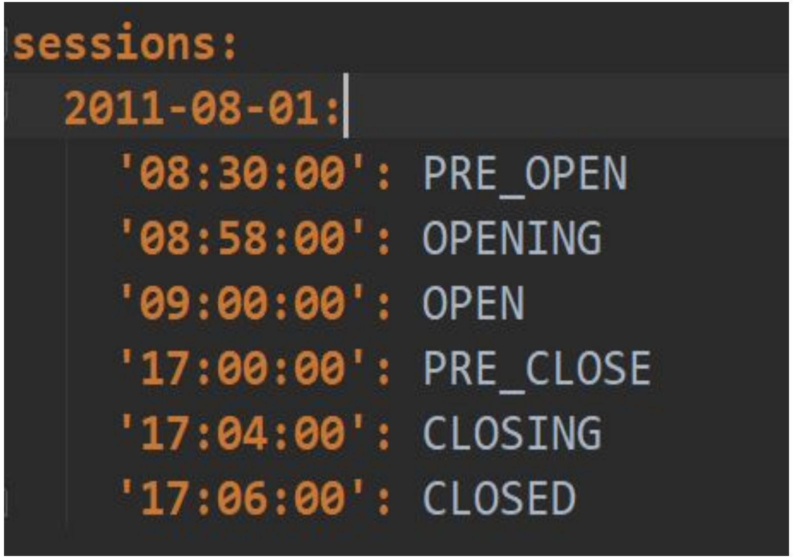
```
download_method:
  - date_from: '1990-01-01'
    method: 'api'
  - date_from: '2016-10-01'
    method: 'exchangebyday'
```

Sessions

You need to define the trading hours and corresponding the market conditions.

SGX AS AN EXAMPLE

1. Pre-open
2. Opening
3. Open
4. Suspended
5. Pre-close
6. Closing
7. Closed



```
sessions:
  2011-08-01:
    '08:30:00': PRE_OPEN
    '08:58:00': OPENING
    '09:00:00': OPEN
    '17:00:00': PRE_CLOSE
    '17:04:00': CLOSING
    '17:06:00': CLOSED
```

Sessions

1. In many cases, we skip opening and closing because they happen within a few microseconds…
2. Get the official trading hours from the exchange
   a. Easy to get it for current trading hours and usually hard to get the historical ones.

3. TRTH data is noisy, it can be different from what officially reported.

   b. Pick one liquidity security and a few days to see whether the data is well-behaved or not.
   c. If there is small discrepancies, allow 1-2 more mins to include all trades. This practice would allow us to report the trade stats correctly

Listing market and currency

Not all securities trading in an exchange are listed in this exchange. In fact, some exchanges only trade securities listed in other exchanges. For example, Chi-X Australia trades only securities on ASX.

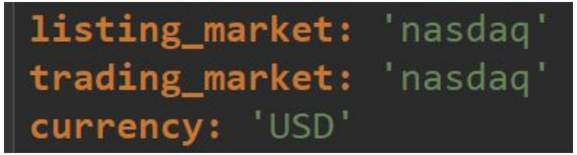
Some markets trade securities in multiple currencies.

We need to know the listing market and the currency of traded securities.

This is done through reading the reference data field of TRTH.

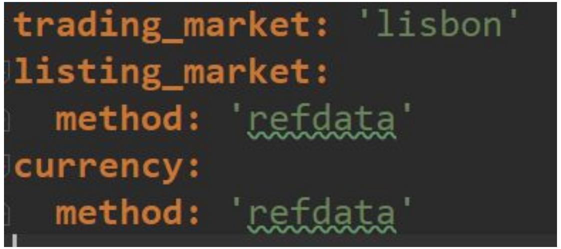
Listing market and currency

If the market is single listed and has only single currency, you can specify the listing market and currency like:



```
listing_market: 'nasdaq'
trading_market: 'nasdaq'
currency: 'USD'
```

If the there is multiple listing markets can trade in the trading venue, and they trade multiple currencies, you need to direct the WorkFlow to read the reference data field from TRTH.



```
trading_market: 'lisbon'
listing_market:
  method: 'refdata'
currency:
  method: 'refdata'
```

Instrument types

This parameter is to define which asset class to download. In TRTH, type 113 is equity type 112 is funds, 97 is equity-linked, and 96 is fund-linked. There are other types which I do not cover in this session (to be covered in session 5).

There are often misclassifications by TRTH, they put equity in fund type and vice versa. Therefore, we may need to download across multiple asset classes to get all the things we need.

We will confirm what to download when we look at the valid security list.

Qualifiers

The qualifiers are attached each transaction to signal the type of the transaction, for example, it can tell you whether the trade is on or off market, whether it is a dark trade or some other types.

Generally, we follow the guide provided by TRTH (This will be shared with you), and record them in the yaml file.

However, there are some qualifiers not covered in TRTH documentation, the convert job will fail if it happens.

Then we will need to look at those missing qualifiers and decide what to do with them.

Listing_market.yaml

Step away from the stream yaml file

Listing_market.yaml defines the market as a listing market.

You will need to fill in the following details:

1. Country - a two-digit country code of where the exchange is located
2. security_code: Use RIC or ISIN to identify security
3. Mic_names - Market identifier code name for the market.
4. Listing_filter - A series of arguments to filter stocks during sync step.

<< Demonstrate listing market yaml file >>

trading_market.yaml

Trading market yaml file defines the market as the trading market.
The parameters:

1. listing_markets: Defines all the listing markets can trade their securities here
2. streams: lists all the forming streams
3. stream_indices: The major index of this trading markets.
4. security_suffixes: lists all the possible security suffix (the part after . (dot))
5. Timezone: the time zone information.
6. Listing filters - A series of arguments to filter stocks during sync step.

<< Demonstrate trading_market.yaml >>

Public holidays and trading weekends

Public holidays are skipped when processing the data for particular exchange. You will need to define them in the market configuration file.

Weekends are automatically skipped when processing the data. If you have certain trading weekends in the market, you will need to include them in the market configuration file for WorkFlow to process them.

<< Demonstrate public holidays and trading weekends >>

tracks. yaml

Tracks serve as a link between a trading market and a number of streams. Tracks are configured in tracks.yaml, and have the form:

```
nasdaq_taq:
  uptick_name: nasdaq
  stream_names: [NAS, NMS, NSM, NBA, NBN]
```

Here ‘nasdaq’ is the name of the trading market, ‘taq’ is the track type (taq = Trades and Quotes), and the streams are NAS, NMS, etc. We have separate entries for all the other track types (e.g. info, depth, corpactions etc) which can contain their own specific sets of streams.

tracks. yaml

You can specify a change in streams over time using a date range:

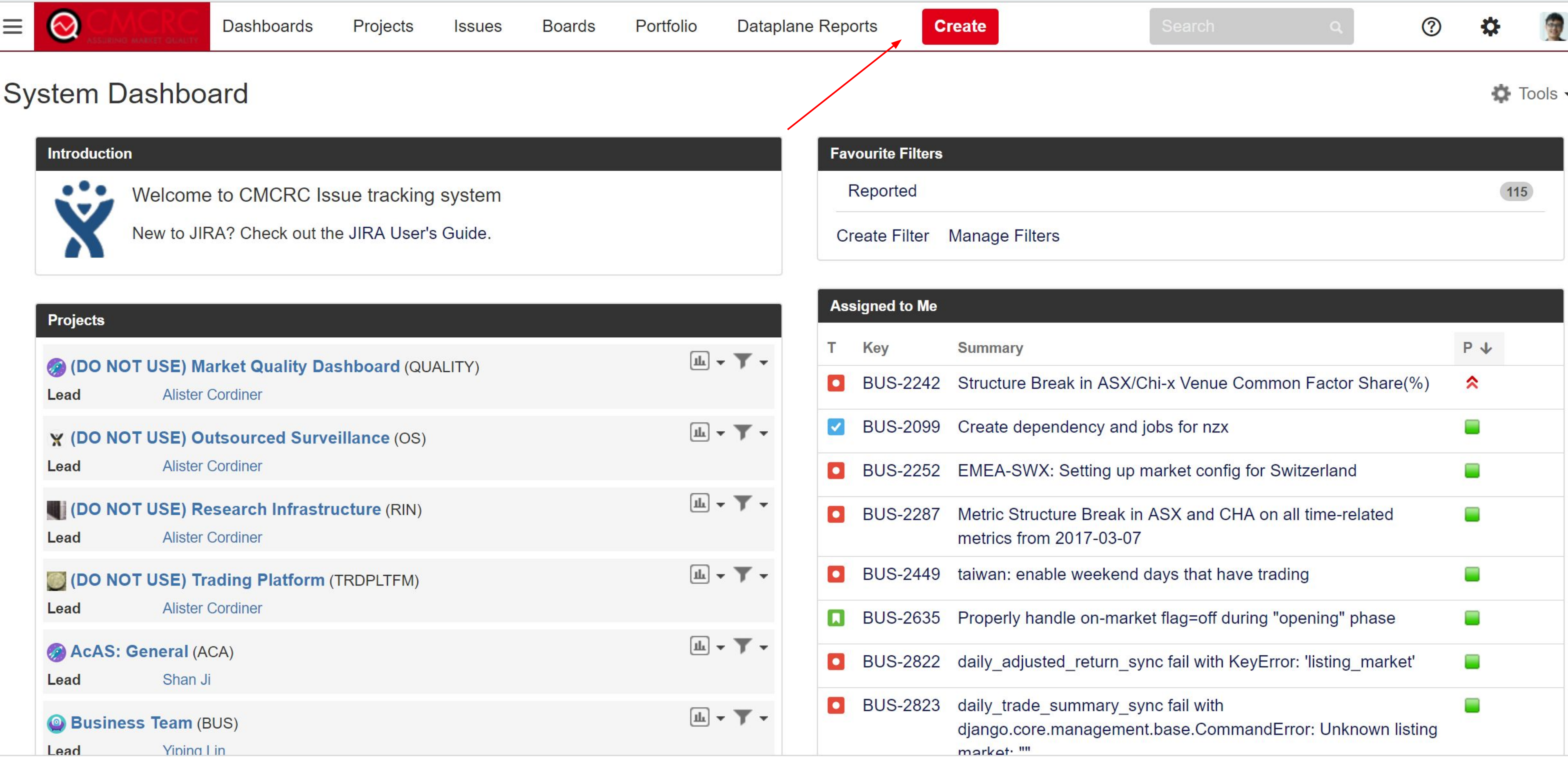
```
nasdaq_taq:
  uptick_name: nasdaq
  stream_names:
    2007-01-01: [NAS, NMS, NSM, THM]
    2015-09-01: [NAS, NMS, NSM, NBA, NBN]
```

<< Brief demo of tracks.yaml>>

<< Demonstrate JIRA, Bitbucket, and how to make changes to code repos >>

How to check out the repository from bitbucket (using git).

https://wiki.cmcrc.com/display/MQ/Check+out+a+repository+from+Bitbucket



System Dashboard

Introduction

Welcome to CMCRC Issue tracking system

New to JIRA? Check out the JIRA User's Guide.

Favourite Filters

Reported 115

Create Filter Manage Filters

Projects

(DO NOT USE) Market Quality Dashboard (QUALITY)
Lead Alister Cordiner

(DO NOT USE) Outsourced Surveillance (OS)
Lead Alister Cordiner

(DO NOT USE) Research Infrastructure (RIN)
Lead Alister Cordiner

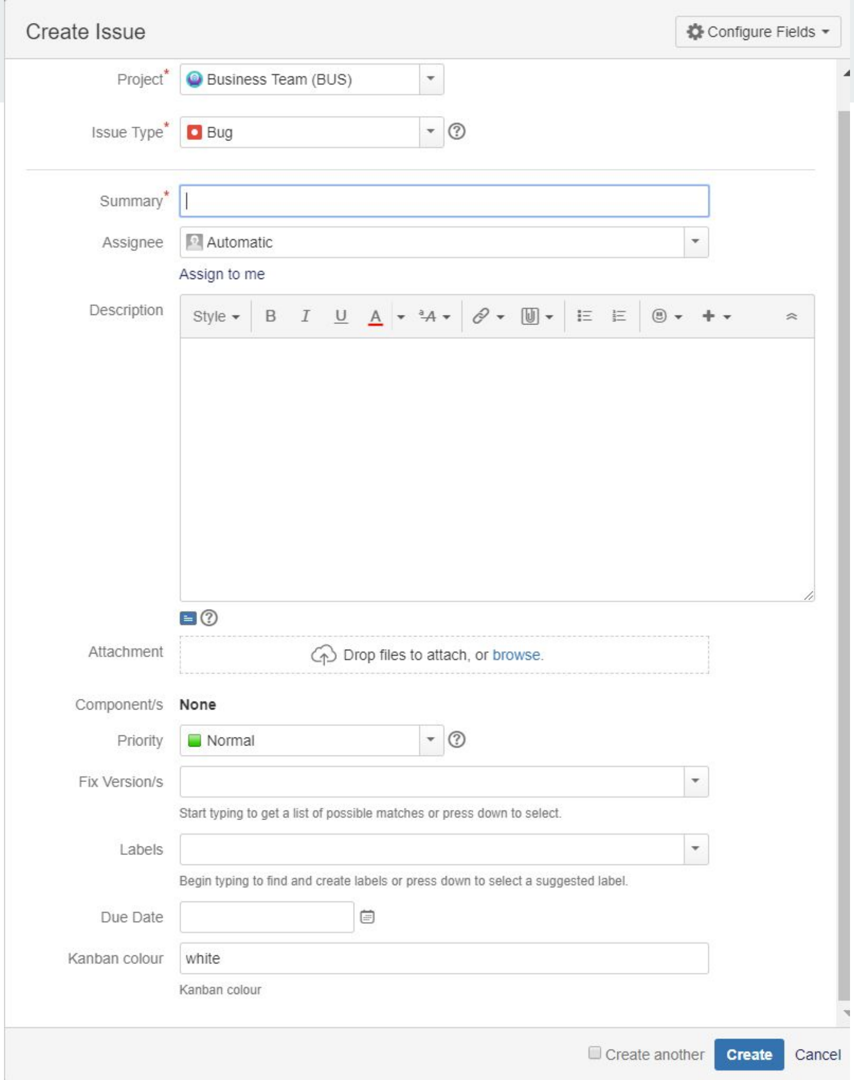
(DO NOT USE) Trading Platform (TRDPLTFM)
Lead Alister Cordiner

AcAS: General (ACA)
Lead Shan Ji

Business Team (BUS)
Lead Yiping Lin

Assigned to Me

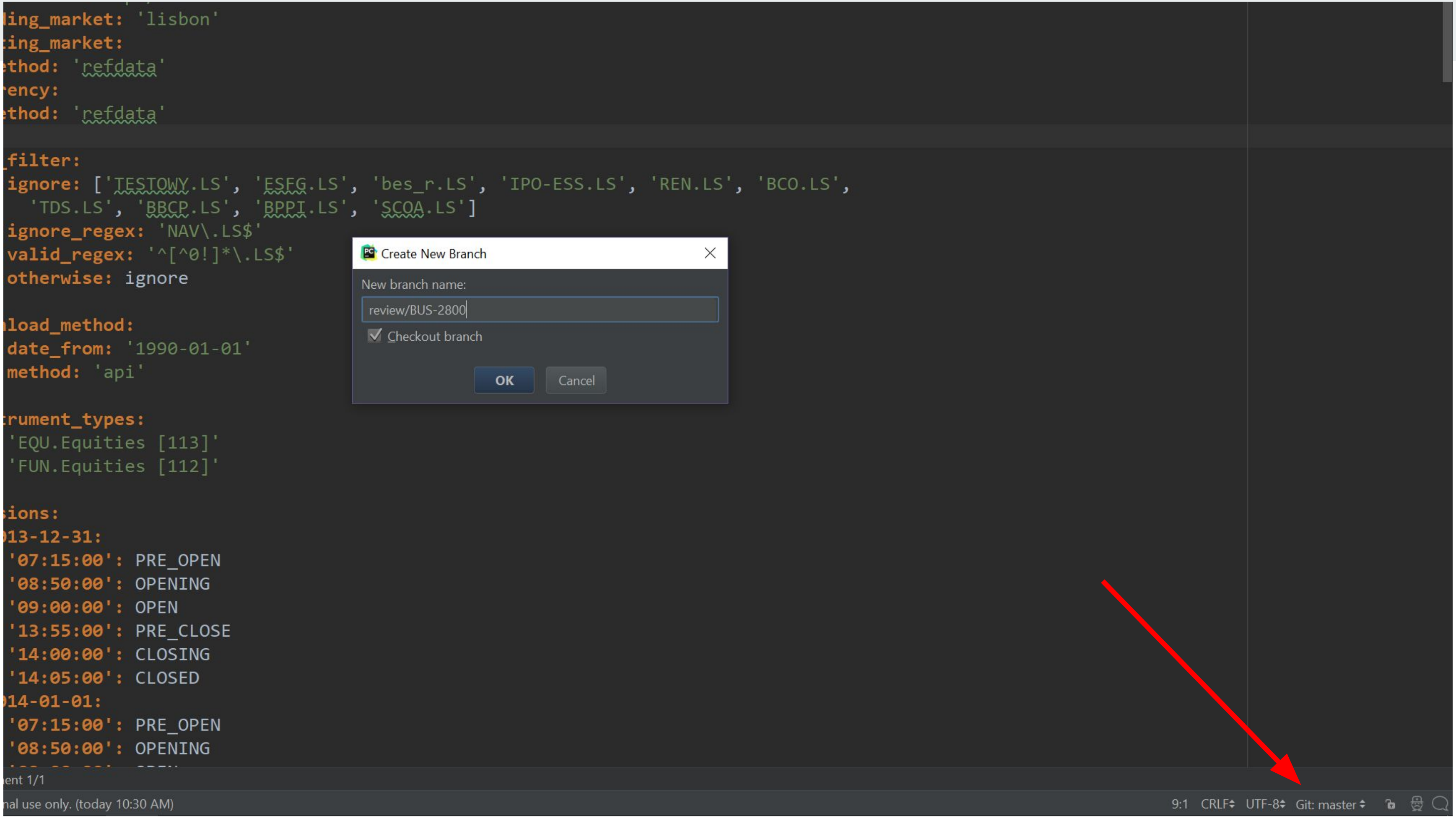
T	Key	Summary	P ↓
🟥	BUS-2242	Structure Break in ASX/Chi-x Venue Common Factor Share(%)	⇈
☑	BUS-2099	Create dependency and jobs for nzx	🟩
🟥	BUS-2252	EMEA-SWX: Setting up market config for Switzerland	🟩
🟥	BUS-2287	Metric Structure Break in ASX and CHA on all time-related metrics from 2017-03-07	🟩
🟥	BUS-2449	taiwan: enable weekend days that have trading	🟩
🟩	BUS-2635	Properly handle on-market flag=off during "opening" phase	🟩
🟥	BUS-2822	daily_adjusted_return_sync fail with KeyError: 'listing_market'	🟩
🟥	BUS-2823	daily_trade_summary_sync fail with django.core.management.base.CommandError: Unknown listing market: ""	🟩



Create Issue

Project*	Business Team (BUS)
Issue Type*	Bug
Summary*	
Assignee	Automatic Assign to me
Description	
Attachment	Drop files to attach, or browse.
Component/s	**None**
Priority	🟩 Normal
Fix Version/s	 Start typing to get a list of possible matches or press down to select.
Labels	 Begin typing to find and create labels or press down to select a suggested label.
Due Date	
Kanban colour	white Kanban colour

□ Create another



```
ling_market: 'lisbon'
ing_market:
thod: 'refdata'
ency:
thod: 'refdata'

filter:
 ignore: ['TESTOWY.LS', 'ESFG.LS', 'bes_r.LS', 'IPO-ESS.LS', 'REN.LS', 'BCO.LS',
   'TDS.LS', 'BBCP.LS', 'BPPI.LS', 'SCOA.LS']
 ignore_regex: 'NAV\.LS$'
 valid_regex: '^[^0!]*\.LS$'
 otherwise: ignore

load_method:
 date_from: '1990-01-01'
 method: 'api'

rument_types:
 'EQU.Equities [113]'
 'FUN.Equities [112]'

ions:
13-12-31:
 '07:15:00': PRE_OPEN
 '08:50:00': OPENING
 '09:00:00': OPEN
 '13:55:00': PRE_CLOSE
 '14:00:00': CLOSING
 '14:05:00': CLOSED
14-01-01:
 '07:15:00': PRE_OPEN
 '08:50:00': OPENING
```

Create New Branch

New branch name: review/BUS-2800

☑ Checkout branch