# BVC Market

Here we will store the library files of our Market.

## Market Strategy

**Notation:**

- `buy price` := the price of the goods offered to the trader.
- `sell price` := the price of the goods that the market is willing to pay to the trader.
- `price` := exchange rate.

## Initial goods allocation:

Note: our market must be initialized with **new_random()** any other use will probably cause unwanted behaviour.

- `eur` := random percentage in range `[25%,35%)` of `STARTING_CAPITAL`.
- `second_good` := random percentage in range `[30%,36%)` of (`STARTING_CAPITAL - eur)`.
- `third_good` := random percentage in range `[45%,55%)` of (`STARTING_CAPITAL - eur - to_eur(second_good)` ).
- `fourth_good` := the remaining capital.

Note: In the initialization, good order is randomized.

## Price fluctuation:

**Premise**: `eur` always has a 1:1 conversion rate

- Define a value `mean` := for each good that is not EUR, convert it to EUR ( using default exchange rates ), sum them all together, then divide the obtained value by 3 ( which is the number of goods except EUR).

$$mean = \frac{\sum_{i=0}^{2} toEur(goods_i)}{3}$$

The following rules are applied:

- If a good has its quantity below the `mean` , (buy) price will fluctuate incrementally using this formula:

$$price = \left(\left(\left(1.0 - \frac{toEur(goodQty) - (toEur(initialGoodQty) \cdot 0.25)}{mean - (toEur(initialGoodQty) \cdot 0.25)}\right) \cdot 0.1\right) + 1.0\right) \cdot defaultPrice$$

This means that if a good it's at its minimum quantity, the trader will pay it 10% more than the default exchange rate.

- If a good has its quantity between `[0%,5%)` over the `mean`, then the default price of that good will be used.

- If a good overcomes the mean by a percentage in range `[5%,10%)`, a favorable price will be applied, hence it will deflate by `2%` from the default price.

- If a good overcomes the mean by a percentage in range `[10%,30%)`, a favorable price will be applied, hence it will deflate by `2.5%` from the default price.

- If a good overcomes the mean by a percentage in range `[30%,60%)`, a favorable price will be applied, hence it will deflate by `3%` from the default price.

- If a good overcomes the mean by more than `60%`, a favorable price will be applied, hence it will deflate by `3.5%` from the default price.

- The `sell price` is always lower than the `buy price`, by exactly `1%`.

- If the trader wants to buy a quantity in range `[25%,30%)` of a certain good, the market will apply a `1%` discount on the `buy price` indiscriminately.

- If the trader wants to buy a quantity in range `[30%,40%)` of a certain good, the market will apply a `1.5%` discount on the `buy price` indiscriminately.

- If the trader wants to buy a quantity in range `[40%,50%)` of a certain good, the market will apply a `2.5%` discount on the `buy price` indiscriminately.

- If the trader wants to buy more than `50%` of a certain good, the market will apply a `3.5%` discount on the `buy price` indiscriminately.

## Locks

The following rules are applied:

- The market refuses **any `lock buy`** that would leave it with less than `25%` of the initial quantity of the asked good.

- The market refuses **any `lock sell`** that would leave it with less than `20%` of the initial `eur` quantity.

- The market will allow to keep a maximum of **4** locks on `lock buy` actions and **4** locks on `lock sell` actions simultaneously

- Locks will expire after **12** days, so that a trader can open at most **4** between buy and sell lock transactions over 3 different markets.

## Good conversion:

The logic is trying to equalize good quantities, but not always, to avoid conflicts with the discount logic applyed in the price fluctuation.

Every time the trader interacts with our or other markets by using `lock sell`, `lock buy`, `buy`, `sell`, there is a `10%` probability that the market will **try** to be rebalance its good quantities among the goods.

How ?

- First compute this value:

$$mean = \frac{\sum_{i=0}^{3} toEur(goods_i)}{4}$$

- Then the market will look for two goods, one that is `suffering` which means it's the one that is worth the less converted to euros and it's not **exported**, and the `chosen good` that is worth the most converted to euros and such that:

$$chosenGood = \max(toEur(good) \; \forall good :$$

$$toEur(good) > toEur(sufferingGood) \wedge notImported(good)$$

Mark the `suffering good` as **imported** and the `chosen good` as **exported** > Note: The eur good will never be marked as Imported or Exported.

Then take a part of the latter, which is:

$$convertedPart = toSufferingGoodKind($$

$$min(mean - toEur(sufferingGood), toEur(chosenGood) - mean))$$

and then sum it to the `suffering` good.

Note: every 24 days the market will reset the Exported/Imported status for each good.

## Event reaction

To keep the market strategy safer and more stable we decided to avoid reacting to external events generated by other markets.