

---

# Brain Documentation

*Release 0.1.0*

Huseyin Uslu

Nov 17, 2017



---

## Contents:

---

|          |                           |          |
|----------|---------------------------|----------|
| <b>1</b> | <b>Contents</b>           | <b>3</b> |
| 1.1      | About . . . . .           | 3        |
| 1.2      | Strategies . . . . .      | 4        |
| <b>2</b> | <b>Indices and tables</b> | <b>7</b> |



**The Brain** is tool for crypto-currency markets aiming to help with users trading activities. It features programmatic, config based and on-the-fly strategies, data and opportunity analyzers.



## 1.1 About

### 1.1.1 Why?

As trading hundreds of cryptocurrencies in tens of markets could be really hard for users, **The Brain** aims to help with decision taking by presenting the user best possible summarized information.

### 1.1.2 How?

**The Brain** consists of multiple parts for use; The UI and Engine. The `Brain Engine` is responsible for running everything behind the curtain including reading & analyzing market data and processing it with opportunity analyzers & strategies.

### 1.1.3 Market Data

**The Brain** can read market data from hundreds of possible cryptocurrency exchanges. Yet again users can also develop their own market data reader programmatically and **The Brain** will be then executing it.

Yet again, **The Brain** can read from fiat conversion API's available through the web to execute cross-fiat conversions.

### 1.1.4 Data Analyzers

Once the market data from all available sources are read, The `Brain Engine` will then will start executing the available data-analyzer modules.

A `data analyzer` is a basic module that can read, process and finally manipulate the data from markets.

As an example `MergedMarketsForPairs` data analyzer merges all currency pairs within the market data and will then find each market with best ask & bid prices.

### 1.1.5 Opportunity Analyzers

Once data analyzers are all processed, The Brain Engine will move on executing opportunity analyzers.

An opportunity analyzers is a basically another type of module that has access to results from data analyzers.

They can basically take the results and check for any outstanding **opportunities** for the current market state.

As an example, a simple opportunity analyzer can read from the previous MergedMarketsForPairs data analyzer's results and calculate if any outstanding opportunities stand for available currency pairs, in which our case buying from the lowest ask market and selling it back in highest bid market would be one.

As well as presenting the user percentage profit, the opportunity analyzer can then create a on-the-fly strategy which will be then later executed and calculate the exact outcome.

### 1.1.6 Strategies

Strategies are basically executed by The Brain Engine to calculated expected outcome & profit for a given set of actions. The Brain Engine can run 3 kind of strategies;

- Configuration based strategies
- Programmatic strategies
- On-the-fly strategies.

## 1.2 Strategies

Strategies are basically executed by The Brain Engine to calculated expected outcome & profit for a given set of actions. The Brain Engine can run 3 kind of strategies;

- Configuration based strategies
- Programmatic strategies
- On-the-fly strategies.

### 1.2.1 Configured

Configuration based strategies are simple YAML markup based strategies. Although it has a pretty basic syntax, it can easily handle the most basic tasks for creating simple strategies.

```
name: TR markets - BTC
description: BTC arbitrage strategy for Turkish markets
region: TR
author: Bonesoul
version: 1.0
enabled: true

defaults:
  input: &input 1000 TRY # default input amount

variants:
  - name: Koinim > Paribu
```

```

description: Buy BTC [Koinim] > Sell BTC [Paribu]
exec:
  - cashin: # Cash in TRY to Koinim.
  - market: Koinim
  - amount: *input
  - buy: BTC # Buy BTC using TRY.
  - transfer: Paribu # Transfer Btc to Paribu.
  - sell: TRY # Sell BTC for TRY.
  - cashout: # Cash out TRY.

```

The above strategy will basically execute these actions;

- The above strategy will basically cash in 1000 USD to Bitcoin exchange called Koinim.
- Will buy Bitcoin with all it's fiat.
- Transfer Bitcoins to another exchange called Paribu.
- Will sell all the Bitcoins.
- Will cash out from the market.

## 1.2.2 Programmatic

Programmatic strategies are custom developed strategies by the user with C# or any other .Net based language. They are far more powered compared to Configuration based strategies as the user will be able to develop virtually every possible trading & arbitrage strategy.

As an example, the code below executes the same above strategy;

```

using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using Brain.Framework.Finances.Currencies;
using Brain.Framework.Finances.Generic;
using Brain.Framework.Strategies.Programmatic;
using Brain.Framework.Strategies.Variants;
using Brain.Framework.Strategies.Variants.Programmatic;

namespace Brain.Strategies
{
    [Serializable]
    public class Strategy : IProgrammaticStrategy
    {
        public string Name { get; }

        public string Author { get; }

        public IReadOnlyCollection<IVariant> Variants { get; }

        [NonSerialized]
        private readonly IList<IVariant> _variants;

        private readonly FiatCurrency _try = new FiatCurrency("TRY");

        private readonly CoinCurrency _btc = new CoinCurrency("BTC");

        public Strategy2()
        {

```

```
        Name = "TR markets - BTC";
        Author = "Bonesoul";

        _variants = new List<IVariant>();
        Variants = new ReadOnlyCollection<IVariant>(_variants);
    }

    public void Load()
    {
        var variant = new ProgrammaticVariant(this, "Koinim -> Paribu
↪");

        variant1.Execute = new Action(() =>
        {
            variant.CashIn("Koinim", new Amount(1000, _try));
            variant.Buy(_btc);
            variant.Transfer("Paribu");
            variant.Sell(_try);
            variant.CashOut();

        });

        _variants.Add(variant);
    }
}
```

Yet again it will execute the same actions;

- The above strategy will basically cash in 1000 USD to Bitcoin exchange called Koinim.
- Will buy Bitcoin with all it's fiat.
- Transfer Bitcoins to another exchange called Paribu.
- Will sell all the Bitcoins.
- Will cash out from the market.

### 1.2.3 On-the-fly

On-the-fly strategies are strategies that are not created by the user but The Brain Engine itself based on current market opportunities. Once the market data is read and processed by data analyzers, a opportunity analyzer that can evaluate possible profit opportunities within the markets and if so, it can dynamically create a suitable on-the-fly strategy for it.

The basic workflow for On-the-fly strategies is;

- The Brain Engine will read data from markets.
- Available data analyzers will be executed where they will process the market data.
- Opportunity analyzers will be evaluating the outcome from data analyzers. Once an opportunity is found, it'll present the user percentage profit and create a on-the-fly strategy utilizing it to exactly calculate the expected outcome.

## CHAPTER 2

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`