Box-Link:

PLEASE READ “READ ME” IN RELEASE FOLDER

Coursework Part A

Real Time Programming

“Currency Exchange Prediction”

Due date: 05/03/2017

**Student:**

Name: Sören Schreiber

Kingston ID: K1549360

Module leader: Dr. Vasils Argyriou(VA)

Lecturers: -Dr. Vasilis Argzriou (VA)

Marker: - Dr. Vasilis Argzriou (VA)

**Table of Contents**

[Introduction 3](#_Toc476388009)

[Assignment 3](#_Toc476388010)

[“Currency Exchange Prediction” 4](#_Toc476388011)

[Application Design 4](#_Toc476388012)

[Application Interface 5](#_Toc476388013)

[ Main Application Form 5](#_Toc476388014)

[ Add exchange rate to application 6](#_Toc476388015)

[ Create new Neural Network 6](#_Toc476388016)

[ View Application Statistics 8](#_Toc476388017)

[Source code 8](#_Toc476388018)

[Functions 8](#_Toc476388019)

[Main Form 8](#_Toc476388020)

[New Neural Network 10](#_Toc476388021)

[New Currency 11](#_Toc476388022)

[Application Statistic 12](#_Toc476388023)

[Neural Network 12](#_Toc476388024)

[Network layer 14](#_Toc476388025)

[File IO 14](#_Toc476388026)

[Results 15](#_Toc476388027)

[Annex 16](#_Toc476388028)

[Table of Figures 16](#_Toc476388029)

[Table of Tables 16](#_Toc476388030)

[References 16](#_Toc476388031)

# Introduction

The first assignment for the real-time programming module focuses on all techniques and technologies introduced in the first week of lectures. The goal of this coursework is the creation of a real-time application.

## Assignment

This assignment focuses on creating a real-time application with the purpose of gathering and predicting changes of currency exchange rates. The application will use real-time gathering of information provided by an online currency exchange API, allowing for simultaneous observation of multiple exchanged rates at the same time.

The application will also try to predict the next change of the exchange rate by using neural networks and gathered historic data.

# “Currency Exchange Prediction”

## Application Design

The following pictures show the original design of the application before start of the project coding.

Not all features and windows forms are included within the final application.

|  |  |
| --- | --- |
| Mock-Up design for application forms | Description |
| Figure 1 - Main mock-up | The main form is the heart of the application. Here all the gathered information, including historic data, and predictions will be display.  The user can manipulate multiple key values relevant to the performance of the application. |
| Figure 2 - Add Currency mock-up | This form will be used to add any currency exchange rate to the To-Watch list of the main program. |
| Figure 3 - Application Statistic mock-up | Application statistics are used to show all numbers relating to the prediction part of the application. |
| Figure 4 - History data form mock-up | The historic data form was planned to provide an interface to see and determine relations between multiple currency exchanges.  The user was supposed to be able to select one currency and then see on the graph the historic changes of the exchange rates regarding this currency.  I.E Euro selected and the graph shows Euro-Pound and Euro-Dollar |

Table 1 - Mock-Ups

## Application Interface

### Main Application Form

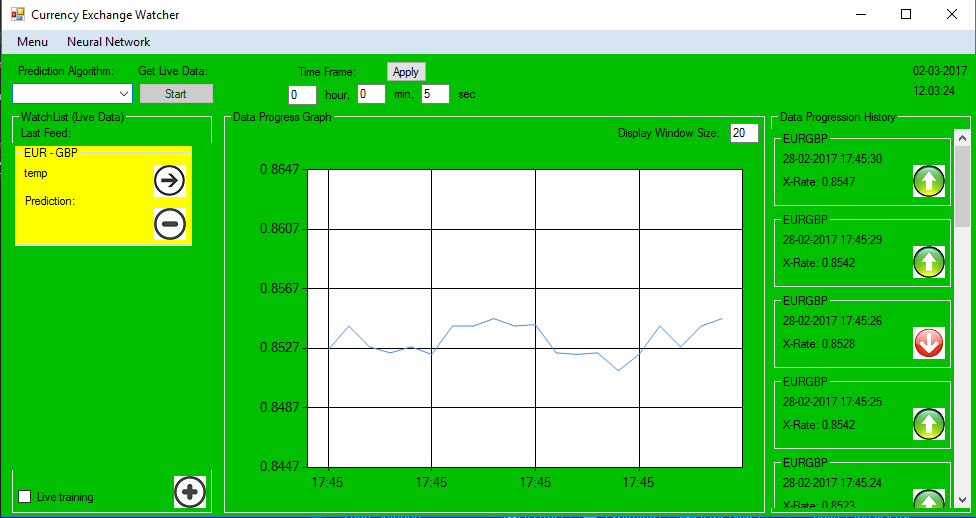


Figure 5 - Main application window

The main application form controls all major application settings and serves as the observe Interface for all selected Currency Exchanges.

The menu drop down will give the user access to the statistics of the application as well a a closing option.

The Neural Network allows any and all control relating to creating, deleting, saving and loading of neural networks.

With the time frame text boxes, the user can control in which interval the application gathers and uses data from the internet.

The user is able to add new currencies by pressing the “+” symbol at the bottom left corner.

With the combo box at the top left and the button named “Start”, the use of a neural network can be selected and the gather of data started.

“Live training” will turn on the retraining algorithm for the current network selected, using the last gathered data from all currently watched currencies.

The entire history data part of the application will be empty on load and only appear and use while a watched currency is being selected by clicking onto the group box containing all information relating to said exchange rate.

The text box named “Display Window Size” will change the amount of data shown in the graph and list. It will update upon leaving the focus on said text box.

### Add exchange rate to application

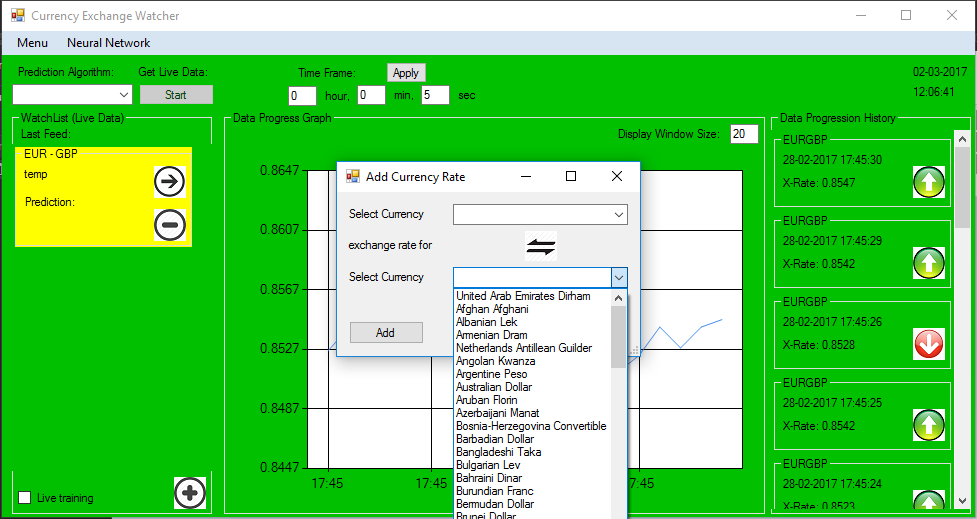


Figure 6 - add currency form

Adding a new currency will be done by using this form of the application.

Any of the Currencies can be selected in either of the selection areas, with the exception of the same one.

The button in the middle will swap selections in both boxes, assuming both have a valid selection.

Cancelling or closing the form will default the form and do nothing, while clicking add will create all relevant and needed information inside the main application.

It will also check if the currency is already being watched.

### Create new Neural Network

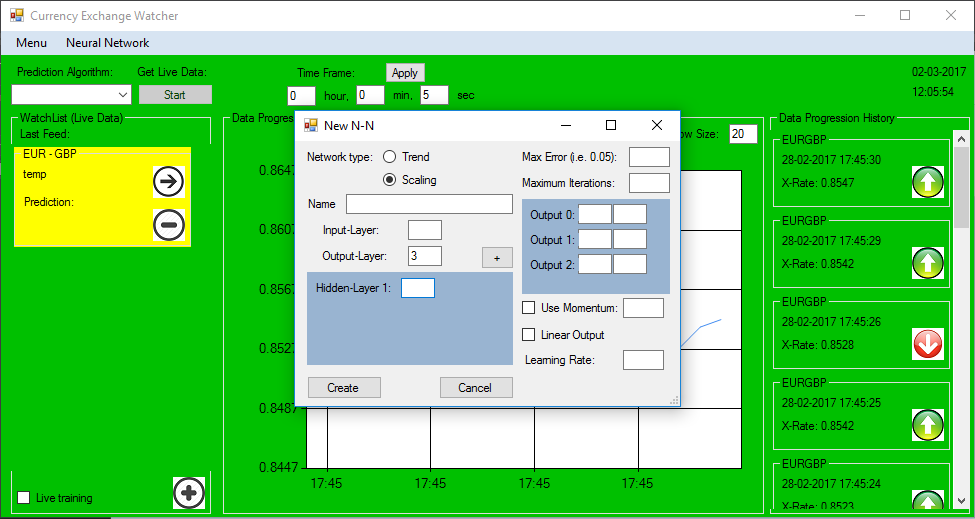


Figure 7 - create new neural network (scale)

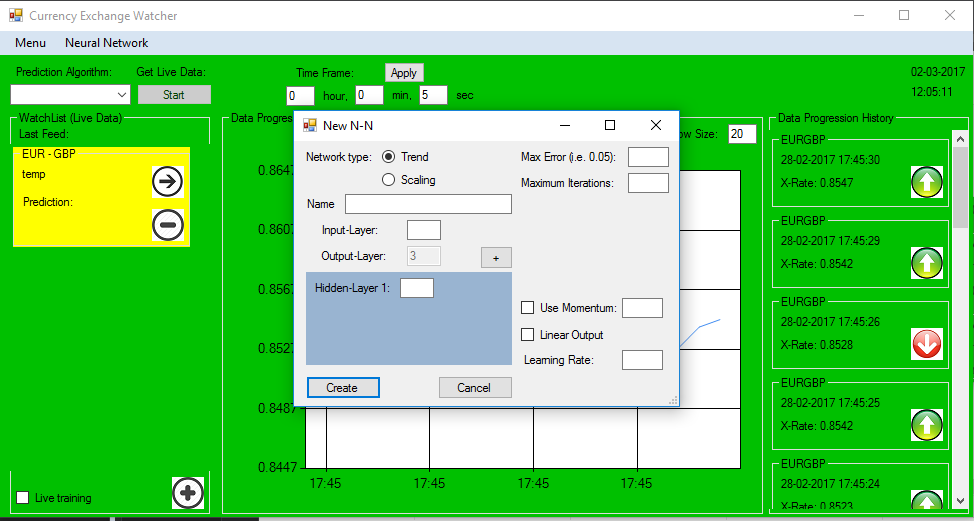


Figure 8 - create new neural network (trend)

This application supports two types of neural networks. All input values will be scaled from -1 and 1 to 0 and 1.

The values Max error and Max Iterations are used for training purposes only.

* Trend:

The user can define any option provided by the interface with the exception of the number of outputs nodes. Those are locked at three for UP(0), LEVEL(1), DOWN(2).

All information, but momentum and linear output, are required to create e new neural network.

The user may specify any number of hidden layers with a minimum of 1.

* Scale:

The scale network allows the user to define all values required for the new neural network.

This kind of network allows the user to set specific values for each of the output nodes. These values will be scaled from 0-1 to the user’s definition.

Leaving the output layer number text box will automatically create required fields on the right-hand side.

### View Application Statistics

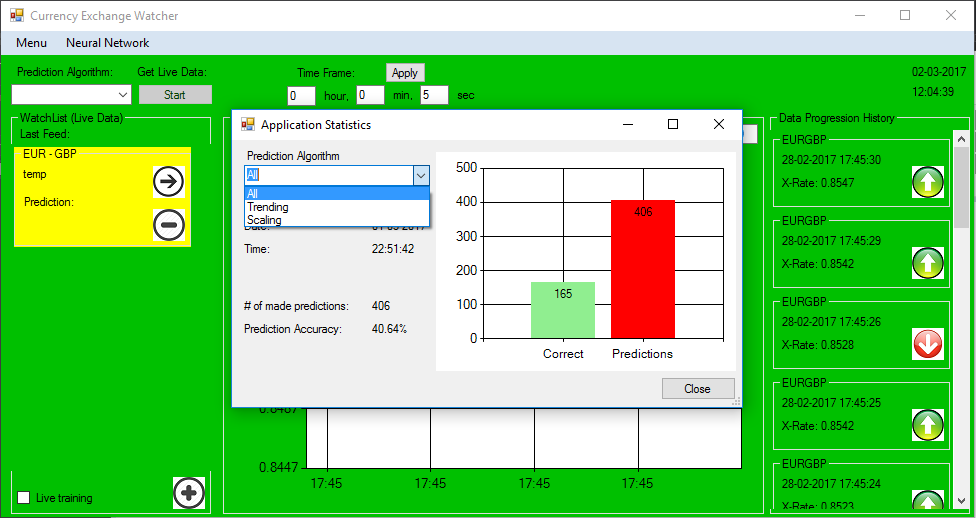


Figure 9 - Application statistics

Application statistics can be accessed through the menu option in the main form.

The statistics show all predictions made, all correct predictions and a percentage value indicating the accuracy of the network type.

Selection options in this application are ALL, TRENDING and SCALING.

# Source code

The next section deals with describing the source code bind the application.

All functions used within the program will be described.

## **Functions**

### Main Form

#### Event Handlers

|  |  |  |
| --- | --- | --- |
| Function | Input | Description |
| timeframe\_KeyPress | System::Object^  System::Windows::Forms::KeyPressEventArgs^ | Checks if pressed key is a digit or backspace |
| MainWindow\_Load | System::Object^  System::EventArgs^ | Sets all default values for the program on start-up and loads up the application statistics and creates a default stats file if no pre-existing one is found |
| TimeFrame\_Tick | System::Object^  System::EventArgs^ | Triggers every user defined time frame to call the live date and update methods |
| time\_Tick | System::Object^  System::EventArgs^ | Triggers every second to display the current date and time |
| button1\_Click | System::Object^  System::EventArgs^ | Starts/Stops TimeFrame timer |
| AddCER\_Click | System::Object^  System::EventArgs^ | Opens the form for adding an exchange rate to the program. And builds the box after currencies are selected |
| deleteWLObject | System::Object^  System::EventArgs^ | Button trigger to delete the box from the watch list |
| loadHDForSelection | System::Object^  System::EventArgs^ | Triggered on exchange box selection. Loads up all historic information for the selected currency exchange |
| textBox1\_Leave | System::Object^  System::EventArgs^ | Applies new time frame to update timer |
| newTool  StripMenuItem  \_Click | System::Object^  System::EventArgs^ | Opens form to create new neural network |
| DeleteNN | System::Object^  System::EventArgs^ | Deletes current selected network from the application |
| deleteNNIncFiles | System::Object^  System::EventArgs^ | Deletes current selected network rom application as well as all application created save files relating to it |
| applicationStatistics  ToolStripMenuItem  \_Click | System::Object^  System::EventArgs^ | Opens the application statistics form |
| saveTool  StripMenuItem1  \_Click | System::Object^  System::EventArgs^ | Saves current network |
| LoadSavedNN | System::Object^  System::EventArgs^ | Loads a network from a selected save file |
| cB\_NeuralNetworks  \_SelectedIndexChanged | System::Object^  System::EventArgs^ | Triggers when new network is selected/ changes all relevant information to match |
| saveAsTool  StripMenuItem  \_Click | System::Object^  System::EventArgs^ | Saves current network in a user specified file |
| MainWindow  \_FormClosing | System::Object^ System::Windows::  Forms::FormClosingEventArgs^ | Saves stats on form close |
| exitTool  StripMenuItem  \_Click | System::Object^ System::EventArgs^ | Closes the application |
| windowSize  \_Leave | System::Object^ System::EventArgs^ | Sets max value of loaded historic data for selected currency |
| trainTool  StripMenuItem  \_Click | System::Object^  System::EventArgs^ | Trains a currently selected network with a user selected training file. |

Table 2 - main::event handlers

#### Dynamic building

|  |  |  |
| --- | --- | --- |
| Function | Input | Description |
| createWactchBox | String^ | This function creates a new dynamic watch boxes for the currency exchange.  This box contains all information relevant to the exchange, like (rate, increase/decrease, prediction, controls) |
| adjustNetworkType | Bool | Adjusts the prediction part of every watch box to match the requirements for the current selection of neural network |
| deleteWatchBox | Control^ | Deletes selected watch box from the container and moves all lower boxes up 1 slot to fill any gaps |
| buildHData | String^ | Loads and displays all historic data up to the window size cap in the history data list and the graph |
| createDPHBox | String^  String^  String^ | Creates and inserts the last live value read to the top of the history data list and graph |

Table 3 - main:dynamic bulding

#### Other Functions

|  |  |  |
| --- | --- | --- |
| Function | Input/Output | Description |
| createSearchPara | Output:  String^ | Creates the search parameters from all the currently watched exchanges |
| getLiveData | Output:  String^  Input:  String^ | Reads and returns live data gathered from the internet, using the search parameters given. |
| runUpdate | Input  Array<String^, 2>^ | The run update function takes data provided by the live data gathering and transform it into the required data types.  Here the prediction is being run to estimate the next change in value. |
| shiftArrayLeft | Input  Array<double>^  double | Shifts a given array 1 slot to the left and insert a new value at the end. |
| plotDPG | Input  DateTime  Double  bool | Draws points onto the graph element of the historic data segment |
| loadStats | - | Loads saved application stats into the main form |
| saveToStats | - | Saves application stats from the main form to the save file |

Table 4 - main:other functions

### New Neural Network

#### Event Handlers

|  |  |  |
| --- | --- | --- |
| Function | Input/Output | Description |
| NewNN\_Load | System::Object^ System::EventArgs^ | Clears and pre-set the form for user use |
| Tb\_name\_Keypress | System::Object^ System::Windows::  Forms::KeyPressEventArgs^ | Changes the back colour to white if the box was marked as false input |
| Textbox1\_keypress | System::Object^ System::Windows::  Forms::KeyPressEventArgs^ | Checks if input is number or backspace.  Also changes back colour if needed |
| DoubleInputControl | System::Object^ System::Windows::  Forms::KeyPressEventArgs^ | Checks if input fits a number, backspace, . or –.  Changes back colour if needed |
| Button1\_Click | System::Object^ System::EventArgs^ | Creates new network and closes the form |
| Button2\_Click | System::Object^ System::EventArgs^ | Closes the form and discards the new network |
| deleteRow | System::Object^ System::EventArgs^ | Deletes hidden input row |
| addRow | System::Object^ System::EventArgs^ | Adds a hidden input row |
| Tb\_output\_leave | System::Object^ System::EventArgs^ | Creates scaling inputs for each output node |
| Rb\_Trend\_CheckedChange | System::Object^ System::EventArgs^ | Changes network type |

Table 5 - new NN:Event handlers

#### Other Functions

|  |  |  |
| --- | --- | --- |
| Function | Input/Output | Description |
| CreateNewNetwork | - | Creates a new neural network with the information provided by the user |
| CreateHiddenArray | - | Creates an array used for parsing hidden layers to the create function |
| FillScaleValues | - | Sets the given scale values for the new network |
| checkInputInformation | Output  bool | Check the form for missing or incorrect inputs |
| CheckNNList | Output  Bool  Input  String^ | Checks if a neural network with the same name exits  False if network name found |
| createHiddenLayerInput | Output  Control^ | Creates a new level of hidden input |
| DeleteHiddenLayerInput | Input  Control^ | Deletes the chosen level of hidden input |

Table 6 - new NN: other functions

### New Currency

#### Event Handlers

|  |  |  |
| --- | --- | --- |
| Function | Input/Output | Description |
| Button2\_Click | System::Object^ System::EventArgs^ | Closes the form and discards user selections |
| Button1\_Click | System::Object^ System::EventArgs^ | Selects currencies for watch list |
| ACR\_Load | System::Object^ System::EventArgs^ | Loads all possible currencies and sets defaults |
| reverseSelection\_Click | System::Object^ System::EventArgs^ | Switches position of both selected currencies |

Table 7 - Add Currency: Event handlers

#### Other fucntions

|  |  |  |
| --- | --- | --- |
| Function | Input/Output | Description |
| generateSearchParameter | Output  String^ | Creates search parameter for selected currency exchange |
| readCurrencyList | - | Gets all currencies from the internet and starts the filling process |
| createCBList | Input  String^ | Reads all currencies and fills the selection boxes and value array |

Table 8 - Add Currency: Other functions

### Application Statistic

#### Event Handlers

|  |  |  |
| --- | --- | --- |
| Function | Input/Output | Description |
| Timer1\_Tick | System::Object^ System::EventArgs^ | Update Statistics every 5 min |
| ApplicationStatistics  \_Load | System::Object^ System::EventArgs^ | Load first read of stats file |
| Close\_Click | System::Object^ System::EventArgs^ | Close form |
| Cb\_NetworkType  \_SelectedIndexChange | System::Object^ System::EventArgs^ | Changes display values |

Table 9 - Application Stats: Event Handlers

#### Other Functions

|  |  |  |
| --- | --- | --- |
| Function | Input/Output | Description |
| loadInformation | - | Load saves statistics from file for display purposes |

Table 10 - Application Stats: other functions

### Neural Network

#### Functions

|  |  |  |
| --- | --- | --- |
| Function | Input/Output | Description |
| INIT | Input  String^  Int  Array<int>  Int  Int |  |
| feedData | - | Feed set Input through the network to get output values |
| PropergateBackwards | - | Change the Network from back to front to adjust for better prediction |
| SetInput | Input  Int  Double | Set specific input node to given value |
| SetWantedOutput | Input  Int  Double | Set wanted value for specific output node with given value |
| SetHiddenLayerNodes | Input  Int  Int  Double | Set the neuron value for a specific node in a specific hidden layer |
| SetBiasValues | Input  Int  Int  Double  Double | Set bias values and weights for a specific node in a specific layer |
| SetWeightValues | Input  Int  Int  Int  Double  Double | Set weight values and weights for a specific node in a specific layer |
| getOuput | Output  Double  Input  Int | Read value from a wanted output node |
| getMaxOutputID | Output  Int | Gets ID of the highest output value node |
| CalculateError | Output  Double | Calculate error value for network |
| setLearningRate | Input  Double | Set the learning rate of the network |
| setLinearOutput | Input  bool | Set linear output for network |
| setMomentum | Input  Bool  Double | Set momentum for network |
| trainNeuralNetwork | Input  NeuralNetwork^  List<array<double>^>  bool | Trains the neural network with given input train data.  Can be used to retrain the network by setting bool to true |
| scaleUp | Output  Double  Input  Double  Double  Double | Scales a given value form 0-1 to given min and max |
| scaleDown | Output  Double  Input  Double  Double  Double | Scales a given Value from min and max to 0-1 |

Table 11 - Neural Network: Functions

### Network layer

#### Functions

|  |  |  |
| --- | --- | --- |
| Function | Input/Output | Description |
| NetworkLayer | - | Constructor for Network layer class |
| INIT | Input  Int  NetworkLayer^  NetworkLayer^ | Initialize the network layer with given information |
| RandWeights | - | Randomize layer weights |
| CalculateError | - | Calculate Layer error |
| AdjustWeights | - | Adjust Layer Weights to adjust for error |
| Calculate Neuron Values | - | Calculate Layer neuron values |
| hasChildLayer | Output  Bool | Checks Layer for existing Child |
| hasParentLayer | Output  Bool | Checks Layer for existing Parent |

Table 12 - Network Layer: Functions

### File IO

#### Functions

|  |  |  |
| --- | --- | --- |
| Function | Input/Output | Description |
| readFromFile | Output  List<String^>  Input  String^  String^ | Read a file line by line and return a list of all lines read |
| writeToFile | Input  String^  String^  String^  Bool | Either appends or overwrites a given file with data string. |
| parseInternetRead | Output  Array<String^,2>  Input  String^  int | Converts data from internet into desired format |
| parseTrainingData | Output  List<array<double>^>  Input  int  String^ | Converts training data file into desired format |
| loadHData | Output  Array<String^,2>  Input  String^ | Loads history data from save file for given currency |
| saveNeuralNetwork | Input  NeuralNetwork^  String^  String^ | Saves selected neural network to a specified file. |
| loadNeuralNetwork | Output  NeuralNetwork^  Input  String^ | Loads a neural network from specified file |
| saveToHData | Input  String^  String^  String^ | Appends new data onto currency history file |
| loadHData | Output  List<String^>  Input  String^  String^ | Reads all lines form given history file and returns them as a list of strings |
| deleteFiles | Input  String^ | Delete given File |
| createFileForTrend | Output  Bool  Input  String^ | Create training data for trend network with a given history file  No access from Application |
| createFileToScale | Output  Bool  Input  String^ | Create training data for scale network with a given history file  No access from Application |

Table 13 - File IO: Functions

# Results

The final application is a great tool to observe currency exchanges live.

It allows for great user input and manipulation of base values and option, while not limiting the user to a certain amount of observable currencies.

The historic data section allows for good and fast observation of past changes of a selected currency exchange rate.

The use of a neural network as a method of predicting the next change, will allow the user to define/ train and use the prediction as they see fit. While the application statistics will give a decent feedback on the performance of the different types of neural networks.

# Annex

## **Table of Figures**

[Figure 1 - Main mock-up 4](#_Toc476387434)

[Figure 2 - Add Currency mock-up 4](#_Toc476387435)

[Figure 3 - Application Statistic mock-up 4](#_Toc476387436)

[Figure 4 - History data form mock-up 4](#_Toc476387437)

[Figure 5 - Main application window 5](#_Toc476387438)

[Figure 6 - add currency form 6](#_Toc476387439)

[Figure 7 - create new neural network (scale) 6](#_Toc476387440)

[Figure 8 - create new neural network (trend) 7](#_Toc476387441)

[Figure 9 - Application statistics 8](#_Toc476387442)

## **Table of Tables**

[Table 1 - Mock-Ups 4](#_Toc476387443)

[Table 2 - main::event handlers 9](#_Toc476387444)

[Table 3 - main:dynamic bulding 10](#_Toc476387445)

[Table 4 - main:other functions 10](#_Toc476387446)

[Table 5 - new NN:Event handlers 11](#_Toc476387447)

[Table 6 - new NN: other functions 11](#_Toc476387448)

[Table 7 - Add Currency: Event handlers 12](#_Toc476387449)

[Table 8 - Add Currency: Other functions 12](#_Toc476387450)

[Table 9 - Application Stats: Event Handlers 12](#_Toc476387451)

[Table 10 - Application Stats: other functions 12](#_Toc476387452)

[Table 11 - Neural Network: Functions 13](#_Toc476387453)

[Table 12 - Network Layer: Functions 14](#_Toc476387454)

[Table 13 - File IO: Functions 15](#_Toc476387455)

## **References**

* Yahoo web API
* https://openexchangerates.org/api/currencies.json