Damian Przesław

Aplikacja monitorująca przepływ środków na giełdę kryptowalut.

1. Wstęp

Niniejsza dokumentacja opisuje projekt aplikacji, której zadaniem jest monitorowanie wpływów i wypływów Eth oraz Usdt i Usdc z giełd kryptowalutowych, a następnie przesyłanie zebranych informacji na telefon.

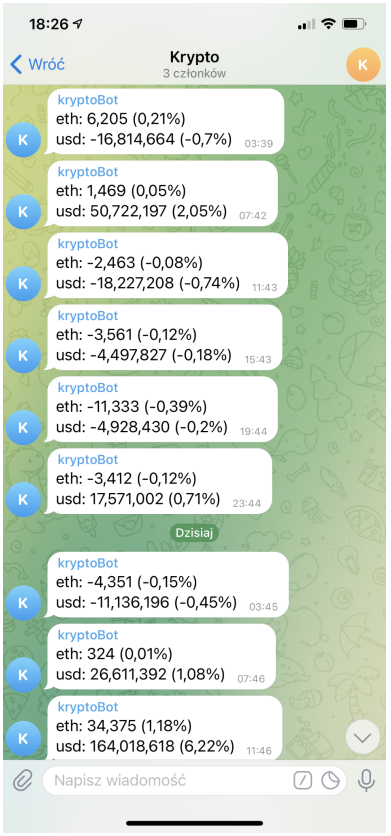
Program został napisany w języku C# w środowisku Microsoft Visual Studio 2019 z wykorzystaniem Windows Form Application na platformie .NET Core

2. Informacje o programie

Program pobiera dane o ilości Eth,Usdt,Usdc portfeli trzech giełd scentralizowanych (Binance, Coinbase, Kucoin) z ethscan.io a następnie sumuje poszczególne wartości i porównuje z poprzednim sprawdzeniem. Program za pomocą bota Telegram wysyła informacje na telefon o zmianie ilości Eth i Usdt. W końcowym etapie wysyła dane do bazy danych.

3. Obsługa programu

Gdy program jest włączony działa przez cały czas wysyłając zebrane informacje na telefon, informacje te można w każdej chwili zobaczyć w aplikacji Telegram w rozmowie z botem.



4. Główna klasa Program

Główne wykonanie programu

```
class Program
{

    Odwołania: 0
    static async Task Main(string[] args)
    {

        connect a = new connect();    // tworzy polaczenie z klientem http ethscan
        Api_adress adress_base = new Api_adress();   // tworzy adresy zapytan
        Bot bot = new Bot();   //tworzy bota telegram
        Values w = new Values();  // wartosci sum danych pobranych
        Date_base db = new Date_base();  //polaczenie z baza danych

        double temp_usdc = 0;     // zmienna do liczenia sumy usd
        double temp_eth = 0;      // zmienna do liczenia sumy eth
        double temp_usd = 0;      // zmienna do liczenia sumy usd

        w.suma_eth_p = db.Pobranie_ostatniej_wartosci_eth();  // pobranie ostatniej zapisanej wartosci sumy eth z bazy danych
        w.suma_usd_p = db.Pobranie_ostatniej_wartosci_usd();  // pobranie ostatniej zapisanej wartosci sumy usd z bazy danych

        while (true) // zapewnia ciagle dzialanie programu
        {
            try
            {
                for (int i = 0; i < 30; i++)    //wysylanie zapytan do kolejnych adresow portweli
                {
                    Console.WriteLine(i);
                    temp_eth = await a.ProcessRepositories_eth(adress_base.get_eth_b(i));    // pobranie ilosci eth z portfela
                    temp_usd = await a.ProcessRepositories_usd(adress_base.get_usdt_b(i));   // pobranie ilosci usdt z portfela
                    temp_usdc = await a.ProcessRepositories_usd(adress_base.get_usdc_b(i));  // pobranie ilosci usdc z portfela

                    w.suma_usd = w.suma_usd + temp_usd + temp_usdc;  //sumowanie usd
                    w.suma_eth = w.suma_eth + temp_eth;              //sumowanie eth

                    Task.Delay(600).Wait();                          // opoznienie zapytania aby nie przekroczyc 5 na 1 sek.
                }
                }

                Console.WriteLine("eth: " + w.suma_eth);
                Console.WriteLine("usd: " + w.suma_usd);

            }
            catch (Exception e)  // wylapanie bledu podczas podczas zapytan
            {
                System.Console.WriteLine(e);
            }

            DateTime thisHour = DateTime.Now;            //pobranie czasu
            Console.WriteLine(thisHour.ToString() + " ");    //wypisanie czasu

            try
            {
                w.cena_eth = await a.ProcessRepositories_eth_price(adress_base.get_cena_eth());  // pobranie ceny eth

            }
            catch (Exception e) // wylapanie bledu podczas podczas zapytania o cene eth
            {
                Console.WriteLine(e);
            }

            bot.send(w); // wyslanie danych na teleram

            db.EthToDateBase(w); // wysłanie danych o eth do bazy danych
            db.UsdToDateBase(w); // wysąlnie danych o usd do bazy danych

            w.next(); //  zapisanie sumy jako poprzedniej i czyszczenie

            Task.Delay(240 * 60 * 1000).Wait();    //opoznienie do nastepnego pobrania danych za 4h
        }
    }
}
}
```

5. Klasy programu

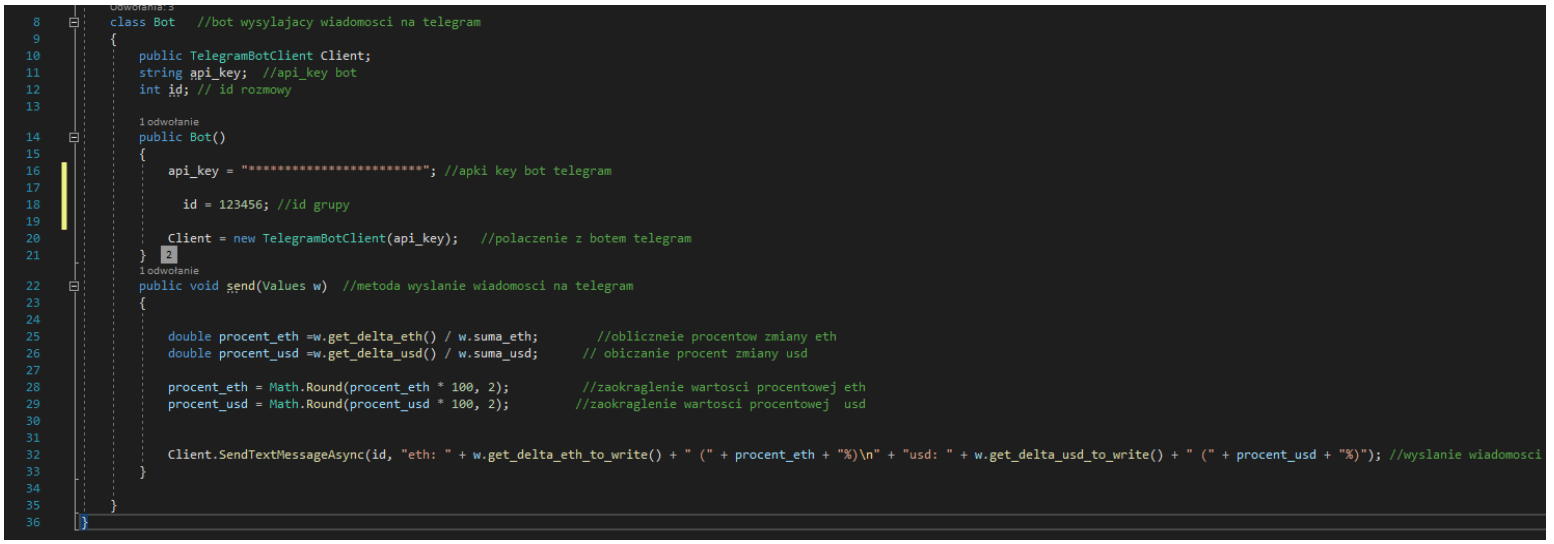
5.1. Api_adress

Klasa posiada potrzebne części adresów do zapytań, które tworzy i zwraca za pomocą metod.

```
Odwołania: 3
class Api_adress  //tworzy adresy zapytan
{
    string[] adres_wallet_b = new string[30]; //adresy binance na sieci eth

    string adres_base = "https://api.etherscan.io/api"; //adres bazowy
    string check_balance = "?module=account&action=tokenbalance"; //sprawdzenie ilosci tokenow

    string eth_price = "?module=stats&action=ethprice";          //sprawdzenie ceny eth

    string adres_usdt = "&contractaddress=0xdac17f958d2ee523a2206206994597c13d831ec7"; //adres usdt
    string adres_usdc = "&contractaddress=0xa0b86991c6218b36c1d19d4a2e9eb0ce3606eb48";  // adres usdc

    string adres_eth = "?module=account&action=balance";// wartosc eth na koncie

    string api_key = "&apikey=******************************"; //api key

    1 odwołanie
    public Api_adress()
    {
        ////////////////////////////////////////////////////////////////////////////////////////// binance
        adres_wallet_b[0] = "&address=0x3f5CE5FBFe3E9af3971dD833D26bA9b5C936f0bE&tag=latest";
        adres_wallet_b[1] = "&address=0xD551234Ae421e3BCBA99A0Da6d736074f22192FF&tag=latest";
        adres_wallet_b[2] = "&address=0x564286362092D8e7936f0549571a803B203aAceD&tag=latest";
        adres_wallet_b[3] = "&address=0x0681d8Db095565FE8A346fA0277bFfdE9C0eDBBF&tag=latest";
        adres_wallet_b[4] = "&address=0xfe9e8709d3215310075d67e3ed32a380ccf451c8&tag=latest";
        adres_wallet_b[5] = "&address=0x4e9ce36e442e55ecd9025b9a6e0d88485d628a67&tag=latest";
        adres_wallet_b[6] = "&address=0xbe0eb53f46cd790cd13851d5eff43d12404d33e8&tag=latest";
        adres_wallet_b[7] = "&address=0xf977814e90da44bfa03b6295a0616a897441acec&tag=latest";
        adres_wallet_b[8] = "&address=0x001866ae5b3de6caa5a51543fd9fb64f524f5478&tag=latest";
        adres_wallet_b[9] = "&address=0x85b931a32a0725be14285b66f1a22178c672d69b&tag=latest";
        adres_wallet_b[10] = "&address=0x708396f17127c42383e3b9014072679b2f60b82f&tag=latest";
        adres_wallet_b[11] = "&address=0xe0f0cfde7ee664943906f17f7f14342e76a5cec7&tag=latest";
        adres_wallet_b[12] = "&address=0x8f22f2063d253846b53609231ed80fa571bc0c8f&tag=latest";
        adres_wallet_b[13] = "&address=0x28C6c06298d514Db089934071355E5743bf21d60&tag=latest";
        adres_wallet_b[14] = "&address=0x21a31Ee1afC51d94C2eFcCAa2092aD1028285549&tag=latest";
        adres_wallet_b[15] = "&address=0xDFd5293D8e347dFe59E90eFd55b2956a1343963d&tag=latest";
        adres_wallet_b[16] = "&address=0x56Eddb7aa87536c09CCc2793473599fD21A8b17F&tag=latest";
        adres_wallet_b[17] = "&address=0x9696f59E4d72E237BE84fFD425DCaD154Bf96976&tag=latest";
        adres_wallet_b[18] = "&address=0x4D9fF50EF4dA947364BB9650892B2554e7BE5E2B&tag=latest";
        adres_wallet_b[19] = "&address=0x4976A4A02f38326660D17bf34b431dC6e2eb2327&tag=latest";
        ////////////////////////////////////////////////////////////////////////////////////////////////////coinbase
        adres_wallet_b[20] = "&address=0x71660c4005ba85c37ccec55d0c4493e66fe775d3&tag=latest";
        adres_wallet_b[21] = "&address=0x503828976d22510aad0201ac7ec88293211d23da&tag=latest";
        adres_wallet_b[22] = "&address=0xddfabcdc4d8ffc6d5beaf154f18b778f892a0740&tag=latest";
        adres_wallet_b[23] = "&address=0x3cd751e6b0078be393132286c442345e5dc49699&tag=latest";
        adres_wallet_b[24] = "&address=0xb5d85cbf7cb3ee0d56b3bb207d5fc4b82f43f511&tag=latest";
        adres_wallet_b[25] = "&address=0xeb2629a2734e272bcc07bda959863f316f4bd4cf&tag=latest";
        ////////////////////////////////////////////////////////////////////////////////////////////////////kucoin
        adres_wallet_b[26] = "&address=0x2b5634c42055806a59e9107ed44d43c426e58258&tag=latest";
        adres_wallet_b[27] = "&address=0x689c56aef474df92d44a1b70850f808488f9769c&tag=latest";
        adres_wallet_b[28] = "&address=0xa1d8d972560c2f8144af871db508f0b0b10a3fbf&tag=latest";
        adres_wallet_b[29] = "&address=0x4ad64983349c49defe8d7a4686202d24b25d0ce8&tag=latest";
    }
    1 odwołanie
    public string get_usdt_b(int i) //zwraca adres pobrania ilosc usdt
    {
        return adres_base + check_balance + adres_usdt + adres_wallet_b[i] + api_key;
    }

    1 odwołanie
    public string get_eth_b(int i) //zwraca adres pobrania ilosci eth
    {
        return adres_base + adres_eth + adres_wallet_b[i] + api_key;
    }
    1 odwołanie
    public string get_usdc_b(int i) //zwraca adres pobrania ilosci usdc
    {
        return adres_base + check_balance + adres_usdc + adres_wallet_b[i] + api_key;
    }
    1 odwołanie
    public string get_cena_eth() //zwraca adres pobrania ceny eth
    {
        return adres_base + eth_price + api_key;
    }
}
}
```

5.2. Bot

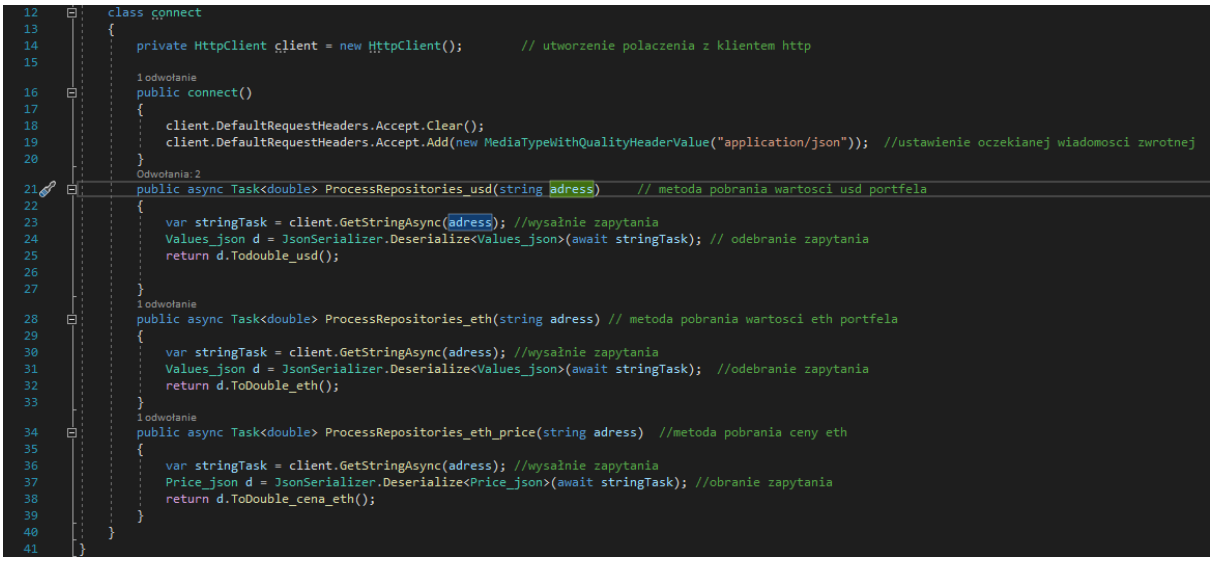
Klasa Bot tworzy bota w aplikacji telegram, który wysyła dane na telefon.



```
Odwołania: 3
class Bot   //bot wysylajacy wiadomosci na telegram
{
    public TelegramBotClient Client;
    string api_key;  //api_key bot
    int id; // id rozmowy

    1 odwołanie
    public Bot()
    {
        api_key = "**********************"; //apki key bot telegram

          id = 123456; //id grupy

        Client = new TelegramBotClient(api_key);   //polaczenie z botem telegram
    }  2
    1 odwołanie
    public void send(Values w)  //metoda wyslanie wiadomosci na telegram
    {

        double procent_eth =w.get_delta_eth() / w.suma_eth;        //obliczneie procentow zmiany eth
        double procent_usd =w.get_delta_usd() / w.suma_usd;      // obiczanie procent zmiany usd

        procent_eth = Math.Round(procent_eth * 100, 2);          //zaokraglenie wartosci procentowej eth
        procent_usd = Math.Round(procent_usd * 100, 2);         //zaokraglenie wartosci procentowej  usd


        Client.SendTextMessageAsync(id, "eth: " + w.get_delta_eth_to_write() + " (" + procent_eth + "%)\n" + "usd: " + w.get_delta_usd_to_write() + " (" + procent_usd + "%)"); //wyslanie wiadomosci
    }

}
}
```

5.3. Connect

Klasa tworzy połączenie z klientem http ethscan.io i za pomoca adresów stworzonych przez klase Api_adress wysyła zapytania i odbiera rezultaty.



```
class connect
{
    private HttpClient client = new HttpClient();        // utworzenie polaczenia z klientem http

    1 odwołanie
    public connect()
    {
        client.DefaultRequestHeaders.Accept.Clear();
        client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));  //ustawienie oczekianej wiadomosci zwrotnej
    }
    Odwołania: 2
    public async Task<double> ProcessRepositories_usd(string adress)     // metoda pobrania wartosci usd portfela
    {
        var stringTask = client.GetStringAsync(adress); //wysałnie zapytania
        Values_json d = JsonSerializer.Deserialize<Values_json>(await stringTask); // odebranie zapytania
        return d.Todouble_usd();


    }
    1 odwołanie
    public async Task<double> ProcessRepositories_eth(string adress) // metoda pobrania wartosci eth portfela
    {
        var stringTask = client.GetStringAsync(adress); //wysałnie zapytania
        Values_json d = JsonSerializer.Deserialize<Values_json>(await stringTask);  //odebranie zapytania
        return d.ToDouble_eth();
    }
    1 odwołanie
    public async Task<double> ProcessRepositories_eth_price(string adress)  //metoda pobrania ceny eth
    {
        var stringTask = client.GetStringAsync(adress); //wysałnie zapytania
        Price_json d = JsonSerializer.Deserialize<Price_json>(await stringTask); //obranie zapytania
        return d.ToDouble_cena_eth();
    }
}
}
```

5.4. Date_base

Klasa odpowiada za komunikację z bazą danych.

```
class Date_base
{
    private MySqlConnectionStringBuilder builder = new MySqlConnectionStringBuilder  // dane logowania do bazy danej
    {
        Server = "*********",
        Database = "*****",
        UserID = "*****",
        Password = "*****",
    };

    1 odwołanie
    public void EthToDateBase(Values w)  // metoda wysyla wartosci eth do bazy danej
    {
        MySqlConnection connect = new MySqlConnection(builder.ConnectionString); //tworzy polaczenie z baza danych

        DateTime thisHour = DateTime.Now;          // pobranie czasu

        string s = string.Format(   // tworzy tresc komendy sql do wstawienia wiersza do bazy
            "INSERT INTO Eth (dzien,godzina,suma,delta,cena) VALUES('{0}','{1}', {2}, {3},{4});",
             thisHour.ToString("yyyy-MM-dd"),
             thisHour.ToString("HH:mm"),
             Math.Round(w.suma_eth),
            Math.Round( w.get_delta_eth()),
             w.get_eth_cena()
             ) ;

        try
        {
            Console.Write("eth ->");
            connect.Open();                    // otwarcie polaczenia z baza danych
            MySqlCommand komenda = connect.CreateCommand();   //  sworzenie komendy sql
            komenda.CommandText = s;        //  przypisanie tresci komendy
          //  komenda.ExecuteReader();        // wykonanie komendy sql
            Console.WriteLine("done");
        }
        catch (Exception e)     // wypisanie bledu z baza danych
        {
            Console.WriteLine("nie udane");
            Console.WriteLine("nie udane");
            Console.WriteLine(e);
        }
    }
    1 odwołanie
    public void UsdToDateBase(Values w)  // metoda wysyla wartosci usd do bazy danej
    {
        MySqlConnection connect = new MySqlConnection(builder.ConnectionString); //tworzy polaczenie z baza danych

        DateTime thisHour = DateTime.Now; // pobranie czasu

        string s = string.Format(     // tworzy tresc komendy sql do wstawienia wiersza do bazy
            "INSERT INTO Usd (dzien,godzina,suma,delta) VALUES('{0}','{1}', {2}, {3});",
             thisHour.ToString("yyyy-MM-dd"),
             thisHour.ToString("HH:mm"),
             Math.Round(w.suma_usd),
            Math.Round(w.get_delta_usd())
            );

        try
        {
            Console.Write("usd ->");
            connect.Open();               // otwarcie polaczenia z baza danych
            MySqlCommand komenda = connect.CreateCommand();    //  sworzenie komendy sql
            komenda.CommandText = s;      //  przypisanie tresci komendy
          //    komenda.ExecuteReader();      // wykonanie komendy sql
            Console.WriteLine("done");


        }

        catch (Exception e)      // wypisanie bledu z baza danych
        {
            Console.WriteLine("nie udane");
            Console.WriteLine(e);
        }
    }
```

```
81          public double Pobranie_ostatniej_wartosci_usd()  //pobiera ostatnia wartosc sumy usd
82          {
83              try
84              {
85                  MySqlConnection connect = new MySqlConnection(builder.ConnectionString); //tworzy polaczenie z baza danych
86                  connect.Open();              // otwarcie polaczenia z baza danych
87
88                  MySqlCommand komenda = connect.CreateCommand();    //  sworzenie komendy sql
89                  komenda.CommandText = "SELECT suma FROM Usd ORDER BY id DESC LIMIT 1;";  //komenda pobrania ostatniej sumy usd
90
91                  MySqlDataReader czytnik = komenda.ExecuteReader();  //wykoanie komendy i owatcie czytnika
92                  czytnik.Read();                    // pobranie wartosci
93
94                  return czytnik.GetInt64(0);
95              }
96              catch (Exception e)
97              {
98                  Console.WriteLine("nie udane pobranie wartosci usd z bazy danych");
99                  return 0;
100             }
101
102         }
```

```
104         public double Pobranie_ostatniej_wartosci_eth() //pobiera ostatnia wartosc sumy usd
105
106         {
107             try
108             {
109                 MySqlConnection connect = new MySqlConnection(builder.ConnectionString); //tworzy polaczenie z baza danych
110                 connect.Open();              // otwarcie polaczenia z baza danych
111
112                 MySqlCommand komenda = connect.CreateCommand();    //  sworzenie komendy sql
113                 komenda.CommandText = "SELECT suma FROM Eth ORDER BY id DESC LIMIT 1;"; //komenda pobrania ostatniej sumy eth
114
115                 MySqlDataReader czytnik = komenda.ExecuteReader(); //wykoanie komendy i owatcie czytnika
116                 czytnik.Read();  // pobranie wartosci
117
118                 return czytnik.GetInt32(0);
119             }
120             catch (Exception e)
121             {
122                 Console.WriteLine("nie udane pobranie wartosci eth z bazy danych");
123                 return 0;
124             }
125         }
126
127     }
128 }
```

5.5. Prince_json

Klasa odpowiada wiadomości zwrotnej o cenie eth w formacie json.

```
8   namespace krypto
9   {
        Odwołania: 2
10      public class Price_json  // klasa odpowiadajaca pobraniu odpowiedzi json o cenie eth
11      {
            Odwołania: 0
12          public string status { get; set; }
            Odwołania: 0
13          public string message { get; set; }
            1 odwołanie
14          public Result result { get; set; }
15
            1 odwołanie
16          public double ToDouble_cena_eth()
17          {
18              return double.Parse(result.ethusd, CultureInfo.InvariantCulture.NumberFormat); // zwrot wartosci ceny eth w double
19          }
20      }
        1 odwołanie
21      public class Result
22      {
            Odwołania: 0
23          public string ethbtc { get; set; }
            Odwołania: 0
24          public string ethbtc_timestamp { get; set; }
            1 odwołanie
25          public string ethusd { get; set; }
            Odwołania: 0
26          public string ethusd_timestamp { get; set; }
27      }
28
29  }
```

5.6. Values

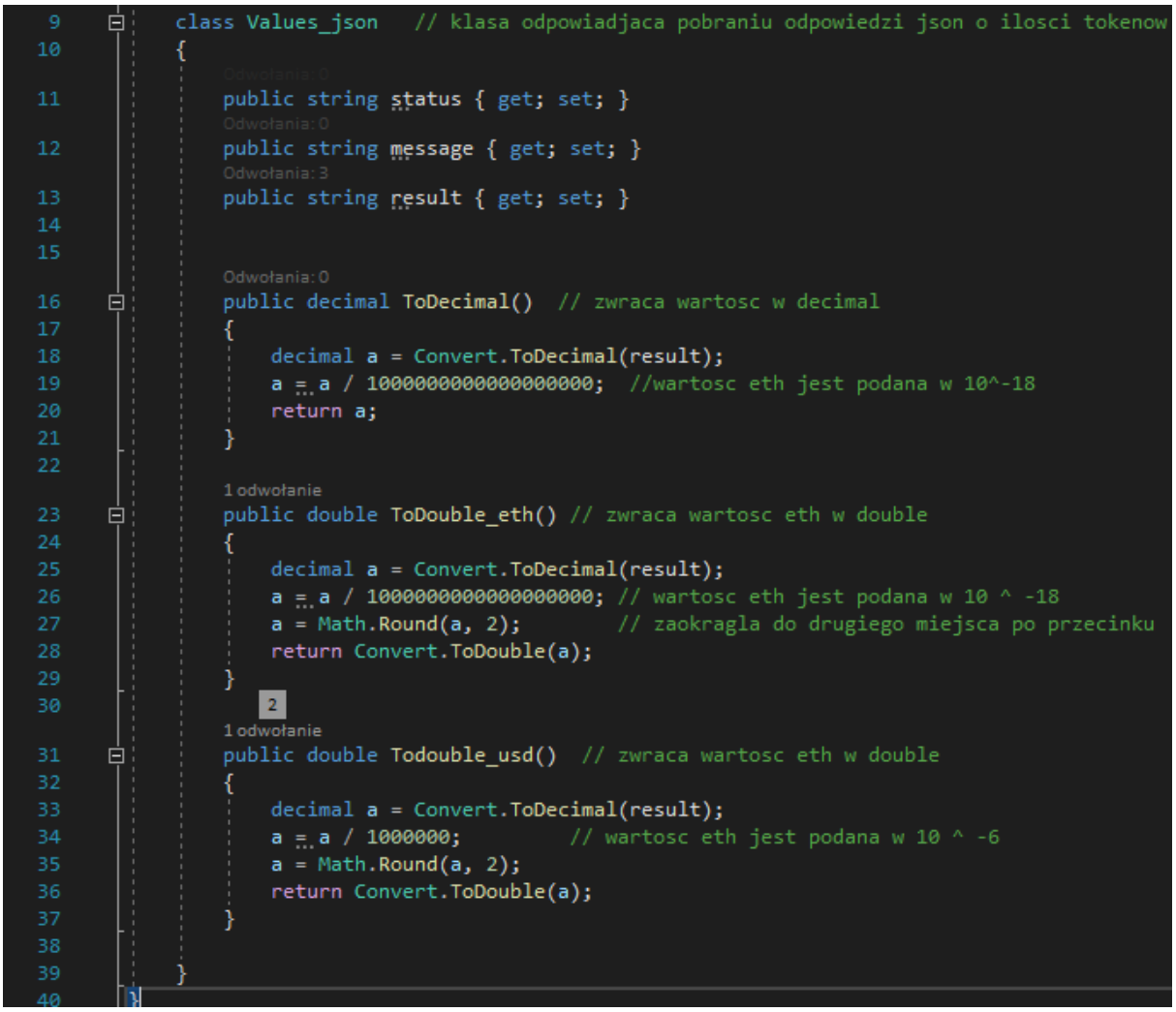
Struktura służy do przechowywania potrzebnych wartości oraz podstawowego przetwarzania do odpowiedniego formatu.

```
10      struct Values  //przechowuje podstawowe wartosci
11      {
12          public double suma_eth;
13          public double suma_usd;
14          public double suma_usd_p;  //porzednia suma usd
15          public double suma_eth_p; // porzednia suma eth
16          public double cena_eth;
17
                Odwołania: 3
18          public double get_delta_eth() // zwraca delte eth
19          {
20              return suma_eth - suma_eth_p;
21          }
                Odwołania: 3
22          public double get_delta_usd() //zwraca delte usd
23          {
24              return suma_usd - suma_usd_p;
25          }
                1 odwołanie
26          public int get_eth_cena() // zwraca zaokraglona cene eth
27          {
28              return Convert.ToInt32(Math.Round(cena_eth));
29          }
                Odwołania: 0
30          public string get_eth_to_write()  //zwraca wartosc eth w odpowiednim formacie do wypisania
31          {
32              return  suma_eth.ToString("0,0", CultureInfo.InvariantCulture);
33          }
                Odwołania: 0
34          public string get_usd_to_write()  //zwraca wartosc usd w odpowiednim formacie do wypisania
35          {
36              return suma_usd.ToString("0,0", CultureInfo.InvariantCulture);
37          }
                1 odwołanie
38          public string get_delta_eth_to_write() //zwraca delte eth w odpowiednim formacie do wypisania
39          {
40              return get_delta_eth().ToString("0,0", CultureInfo.InvariantCulture);
41          }
```

```
42          public string get_delta_usd_to_write() //zwraca delte usd w odpowiednim formacie do wypisania
43          {
44              return get_delta_usd().ToString("0,0", CultureInfo.InvariantCulture);
45          }  2
                1 odwołanie
46          public void next()  // zapisuje sume wartosci jako poprzednia i zeruje sume
47          {
48              suma_eth_p = suma_eth;
49              suma_eth = 0;
50
51              suma_usd_p = suma_usd;
52              suma_usd = 0;
53          }
54      }
55  }
```

5.7. Values_json

Klasa odpowiada odpowiedzi w formacie json co do posiadanych ilości tokenów na portfelu, oraz przetwarza te dane.



```
class Values_json   // klasa odpowiadjaca pobraniu odpowiedzi json o ilosci tokenow
{
    Odwołania: 0
    public string status { get; set; }
    Odwołania: 0
    public string message { get; set; }
    Odwołania: 3
    public string result { get; set; }


    Odwołania: 0
    public decimal ToDecimal()  // zwraca wartosc w decimal
    {
        decimal a = Convert.ToDecimal(result);
        a = a / 1000000000000000000;  //wartosc eth jest podana w 10^-18
        return a;
    }

    1 odwołanie
    public double ToDouble_eth() // zwraca wartosc eth w double
    {
        decimal a = Convert.ToDecimal(result);
        a = a / 1000000000000000000; // wartosc eth jest podana w 10 ^ -18
        a = Math.Round(a, 2);        // zaokragla do drugiego miejsca po przecinku
        return Convert.ToDouble(a);
    }
        2
    1 odwołanie
    public double Todouble_usd()  // zwraca wartosc eth w double
    {
        decimal a = Convert.ToDecimal(result);
        a = a / 1000000;          // wartosc eth jest podana w 10 ^ -6
        a = Math.Round(a, 2);
        return Convert.ToDouble(a);
    }

}
```