МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНСТИТУТ МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ СИСТЕМ
ФАКУЛЬТЕТ АВТОМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ
КАФЕДРА СИСТЕМ АВТОМАТИЗАЦИИ УПРАВЛЕНИЯ

Отчет

по дисциплине

«Научно-исследовательская работа студентов»

Выполнил: _____ /Семакин И.А./
Студент гр. ПИб-3301-01-00 (дата, подпись)
Проверил: _____ /Земцов М.А./
Преподаватель кафедры САУ (дата, подпись)

Киров 2018

1. Обозначение проблемы

В настоящее время существует огромное количество различных социальных сетей и программ для общения. Чтобы просматривать сообщения необходимо заходить в каждое приложение отдельно. Это неудобно людям у которых есть несколько учётных записей в различных социальных сетях и приложениях. Проблема состоит в том, что человеку, имеющему много учётных записей, необходимо больше времени для просмотра сообщений, поскольку ему необходимо каждый раз заходить в соответствующие приложения.

2. Метод решения проблемы

Для решения этой проблемы можно использовать специализированное приложение (мессенджер) которое объединяет все учётные записи в один большой метаконтакт. Больше не нужно будет использовать множество приложений, а только всего одно. Пример такой программы Pidgin. Но в нём отсутствует достаточно популярная социальная сеть «Вконтакте». Мною было принято решение, написать плагин для этой программы чтобы он позволял пользоваться этой социальной сетью из Pidgin.

3. Реализация

В первую очередь, я посетил официальный сайт и прочел информацию. Необходимую для написания плагина.

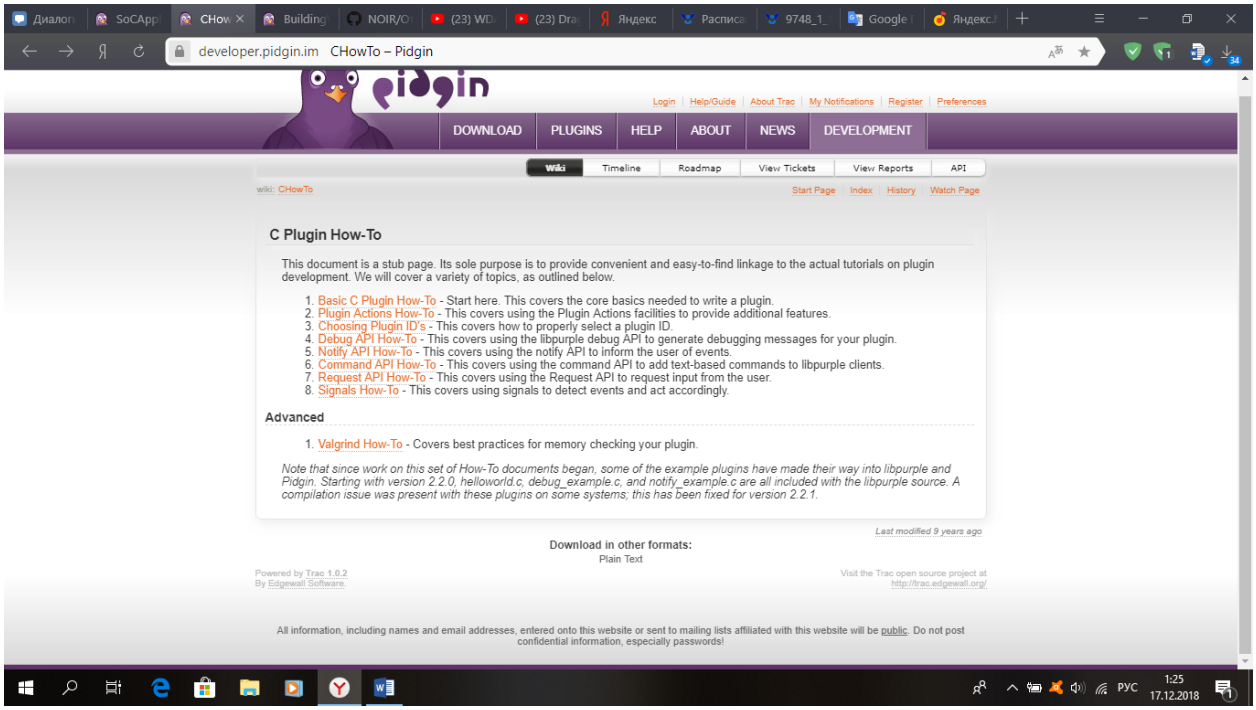

Рисунок 1

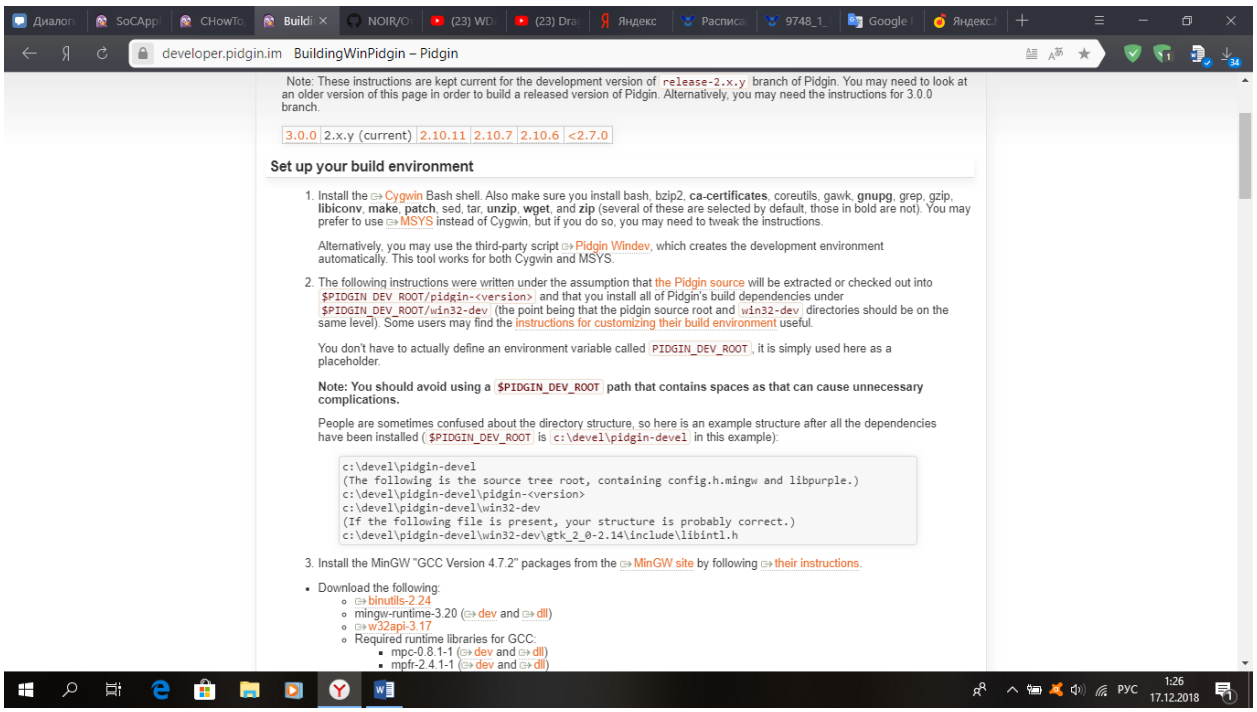

Рисунок 2

Необходимо собрать среду разработки, а именно скачать две программы Cygwin и MinGW, и установить для них необходимые пакеты.

Но сперва я решил потренироваться в работе с API Вконтакте. Что вообще такое API? API (application programming interface) — это посредник между разработчиком приложений и какой-либо средой, с которой это приложение должно взаимодействовать. API упрощает создание кода, поскольку предоставляет набор готовых классов, функций или структур для работы с имеющимися данными.

В среде разработки Visual Studio я написал программу, которая выводит список друзей, и 10 последних сообщений с выбранным другом. Также ему можно отправить сообщение из этой программы.

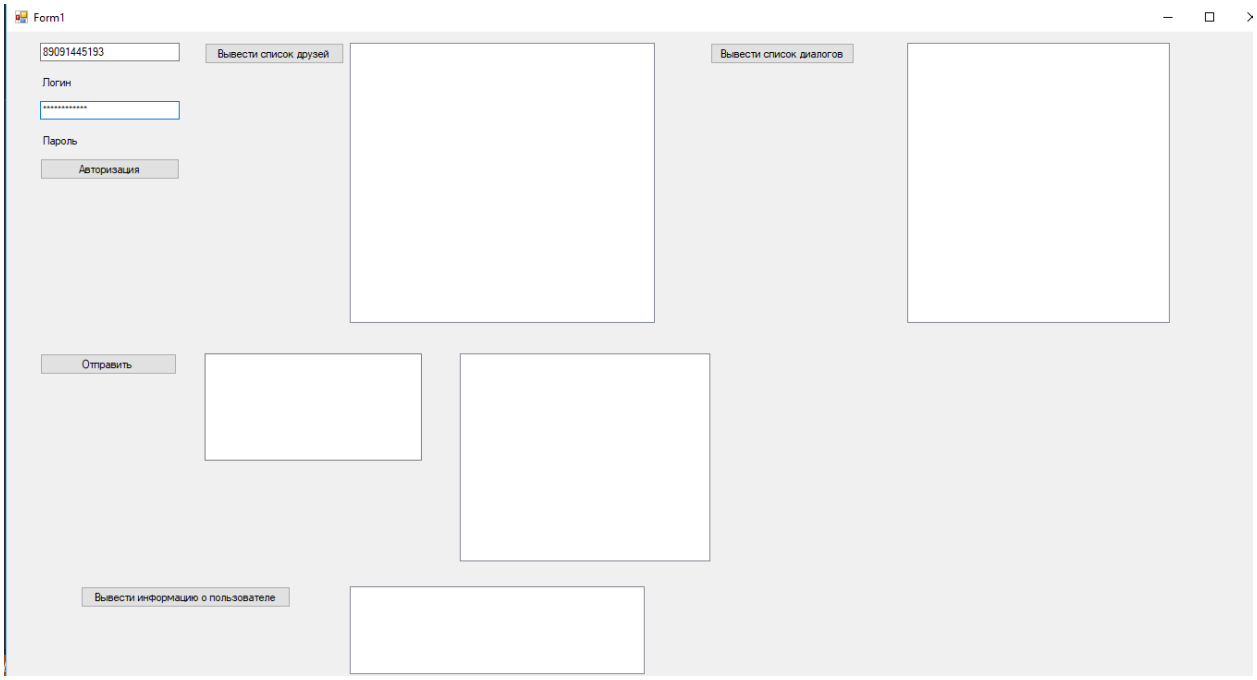

Рисунок 3

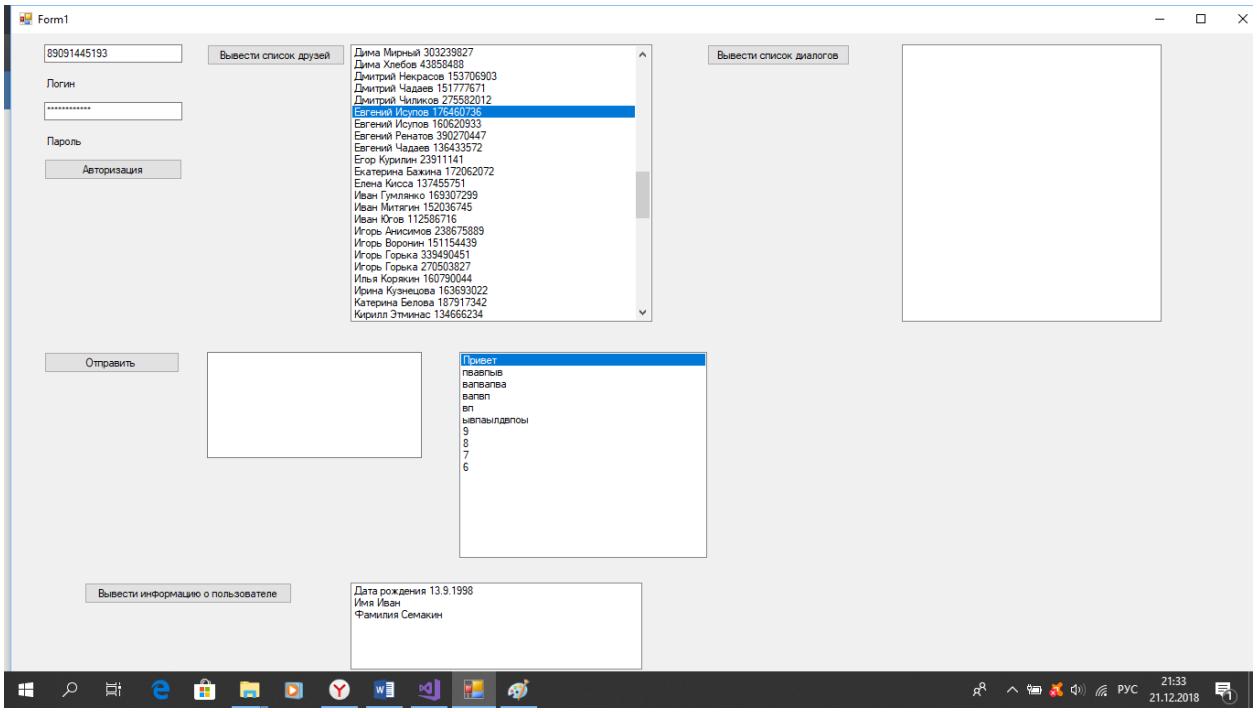

Рисунок 4

Код программы:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using VkNet;
using VkNet.Enums.Filters;
using VkNet.Enums.SafetyEnums;
```

```
using VkNet.Model.RequestParams;

namespace ApiVk
{
    public partial class Form1 : Form
    {
        VkApi vk = new VkApi();
        long UserMessageId = 0;

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {

        }

        private void button1_Click(object sender, EventArgs e)
        {
            ulong appID = 6755556;
            string log = textBox1.Text;
            string pass = textBox2.Text;
            try
            {
                vk.Authorize(new VkNet.Model.ApiAuthParams
                {
                    ApplicationId = appID,
                    Login = log,
                    Password = pass,
                    Settings = Settings.All
                });
                MessageBox.Show("Авторизация прошла успешно");
            }
            catch
            {
                MessageBox.Show("Не удалось авторизоваться");
            }
        }

        public void GetFriends()
        {
            ProfileFields pf = ProfileFields.LastName | ProfileFields.FirstName;
            var fr = vk.Friends.Get(new FriendsGetParams
            {
                Fields = pf,
                Order = FriendsOrder.Name,
            });
            foreach(var fri in fr)
            {
                Listboxesses.Items.Add(fri.FirstName+ " "+ fri.LastName + " " + fri.Id);
            }
        }
        private void button2_Click(object sender, EventArgs e)
        {
            try
            {
                GetFriends();
            MessageBox.Show("Успешно");
        }
            catch
            {
                MessageBox.Show("Не удалось узнать историю сообщений");
            }
```

```
}

        private void button3_Click(object sender, EventArgs e)
        {
            GetProfileInfo();
        }
        public void GetProfileInfo()
        {
            listBox1.Items.Clear();
            var getProfileinfo = vk.Account.GetProfileInfo();
            listBox1.Items.Add("Дата рождения"+" "+getProfileinfo.BirthDate);
            listBox1.Items.Add("Имя"+" "+getProfileinfo.FirstName);
            listBox1.Items.Add("Фамилия"+" "+getProfileinfo.LastName);
        }

        private void Listboxesses_SelectedIndexChanged(object sender, EventArgs e)
        {
            UserMessageId =
long.Parse(Listboxesses.Text.Substring(Listboxesses.Text.LastIndexOf(' ')));
        }

        private void Button4_Click(object sender, EventArgs e)
        {
            try {
                if (textBox3.Text != " ")
                    vk.Messages.Send(new MessagesSendParams
                    {
                        UserId = UserMessageId,
                        Message = textBox3.Text
                    });
                listBox2.Items.Clear();
                var getMess = vk.Messages.GetHistory(new MessagesGetHistoryParams
                {
                    Reversed = false,
                    UserId = UserMessageId,
                    Count = 10
                });
                foreach (var messag in getMess.Messages)
                {
                    listBox2.Items.Add(messag.Text);
                }
                textBox3.Text = " ";
                MessageBox.Show("Успешно");
            }
            catch
            {
                MessageBox.Show("Не удалось");
            }
        }

        private void Button5_Click(object sender, EventArgs e)
        {

        }
    }
  }
```

Затем мною были установлены программы Cygwin и MinGW

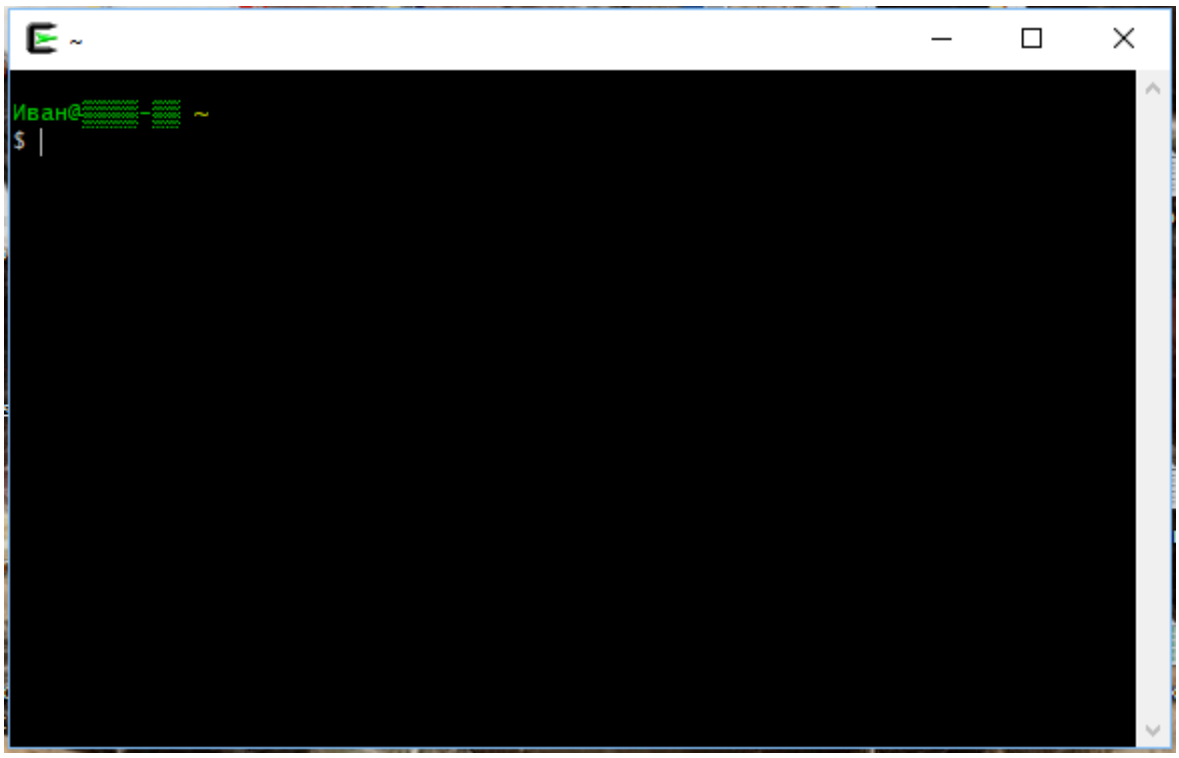

Рисунок 5

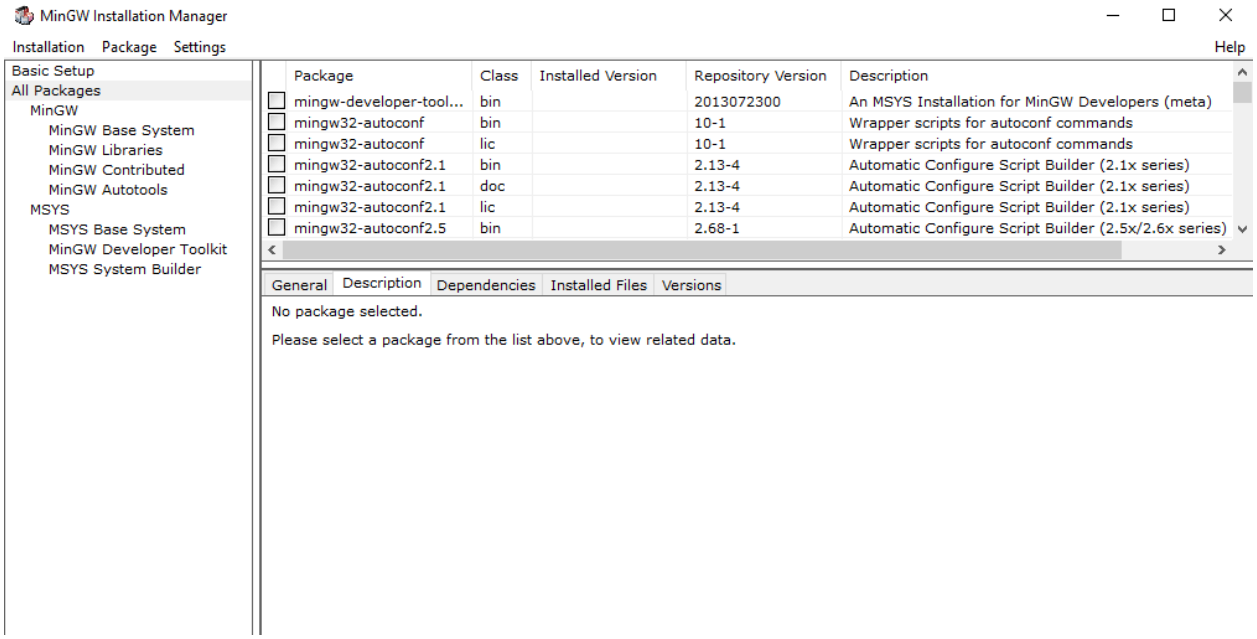

Рисунок 6

Но увы, я не выполнил поставленную перед собой задачу, и не смог написать плагин. Причиной было то что я не смог получить исходный код программы.

Clone the Repository and Select a Branch

To start working with Pidgin from Mercurial, you'll probably want to clone our main repository:

```
hg clone https://bitbucket.org/pidgin/main pidgin
```

This will give you a `pidgin` directory with the 3.0.0 development code by default. If you want to develop against Pidgin 3.0.0, you're all set!

Naturally, Pidgin has a number of branches within its repository, and these branches handle various projects including our maintenance of Pidgin 2.x.y. We won't go into all the branches here, as there are a lot of them, but we know some people would like to work against Pidgin 2.x.y instead of 3.0.0. In order to do that, clone the repository as above, but then do this:

```
cd pidgin
hg up release-2.x.y
```

You now have a Pidgin 2 tree to work with. Happy hacking!

Рисунок 7

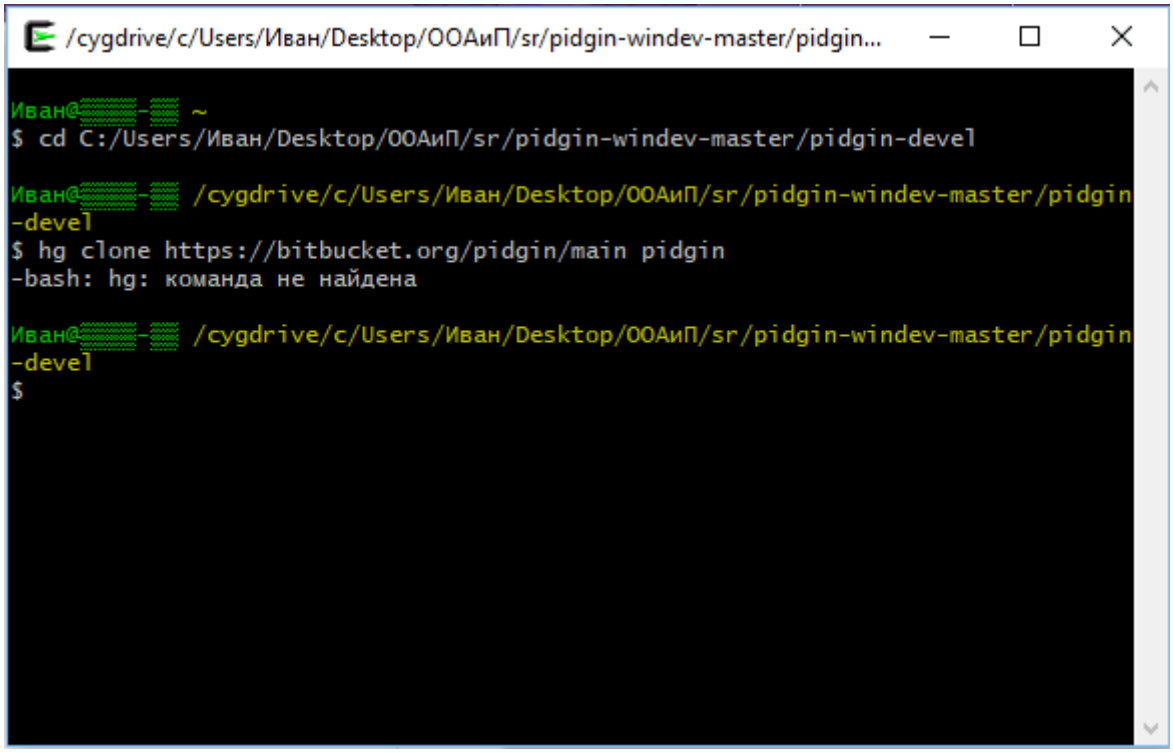

Рисунок 8

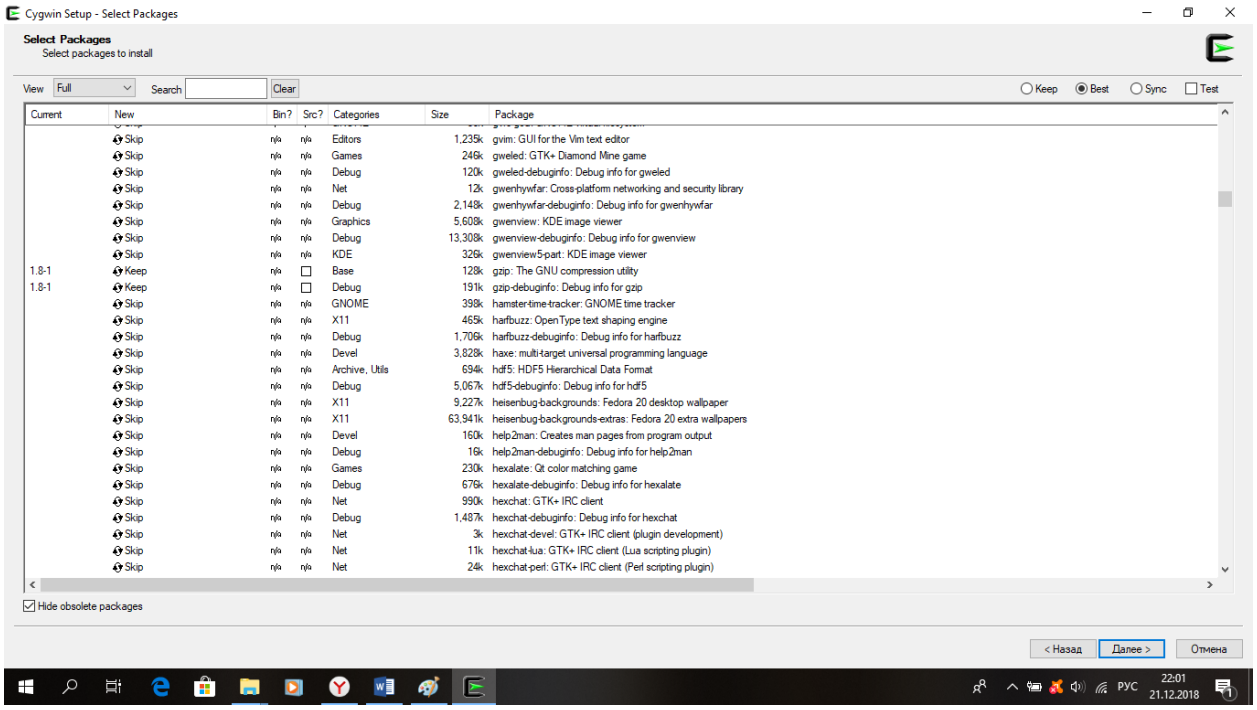

Рисунок 9.

Я не смог найти необходимый пакет, который нужно установить, чтобы эта команда стала доступна.

4. Вывод и итоги

На данный момент моих навыков не хватает для создания плагина. Но в ходе попытки выполнить эту работу, я приобрёл навыки в работе с API Вконтакте. Также выделил для себя, что мне необходимо улучшить знание английского языка, поскольку вся документация связанная с Pidgin на английском языке, как и программы Cygwin и MimGW. Также мне нужно изучить язык C, поскольку Pidgin написан как раз на нём.