Конференция «Использование системы программирования PascalABC.NET в обучении программированию» 2025

Создание Телеграм-ботов на PascalABC.NET – новый взгляд

Пучкин Максим Валентинович

Старший преподаватель каф. прикладной математики и программирования ЮФУ

mpuchkin@sfedu.ru

Почему Телеграм-бот?

Задача: создать систему регистрации для олимпиадной системы PascalABC.NET.

Дано: сервер, проверяющий задания, с базой данных учеников (отработано в течение нескольких лет на школьниках компьютерной школы). Набор скриптов на PHP + база данных MySQL.

Необходима система, позволяющая регистрироваться любому желающему без участия администраторов.

Собственно олимпиады – скачать архив с условиями и заданиями, и с домашнего компьютера решить задачи. Никаких специальных действий не нужно, решения учитываются на сервере автоматически.

Олимпиадная система PascalABC.NET

Олимпиадная система PascalABC.NET

Компьютерная школа мехмата ЮФУ

Южный федеральный университет

Олимпиада: Olymp2-16Feb25 Показать

Период: начало 01.09.2022 00:00 NOW конец 31.05.2025 23:55 Количество записей для показа: 0 Автообновление: Вкл Выкл

Сводка по олимпиаде «Olymp2-16Feb25»:

№	ФИО	Olymp2-Task01	Olymp2-Task02	Olymp2-Task03	Olymp2-Task04	Olymp2-Task05	Olymp2-Task06	Olymp2-Task07	Olymp2-Task08	Olymp2-Task09	Результат
		1	1	1	1	1	1	1	1	1	9
1	Алексеев Андрей Константинович	+0:5	+1:53	+3:8	+17:52	+13:51	+26:43	+17:34	+20:42	+1:56:14	9 8986918
2	Кучер Владислав Алексеевич	+7:31	+8:20	+18:54	+1:36:53	+4:5	+4:52	+5:33	+6:35	+1:17:12	9 8986205
3	Быстров Михаил Евгеньевич	+1:19	+3:10	+10:32	+28:48	+33:51	+37:57	+40:39	+57:17	+1:11:56	9 8982871
4	Колесников Сергей Владимирович	+1:17	+4:36	+10:40	+23:37	+38:2	+42:47	+51:51	+1:50:20	+1:4:6	9 8979164
5	Хоружий Артём Игоревич	+31:30	+18:42	+18:18	+23:7	+36:24	+40:16	+43:28	+1:24:0	+1:8:37	9 8978138
6	Гагарин Константин Юрьевич	+1:16	+4:38	+1:28:7	+22:26	+33:56	+45:30	+50:48	+56:35	+1:18:46	9 8977078

Проблемы

Традиционная схема элементарная: скрипты + отправка почты. Забыли пароль, восстановить доступ – через почту. Отправка почты из PHP элементарная.

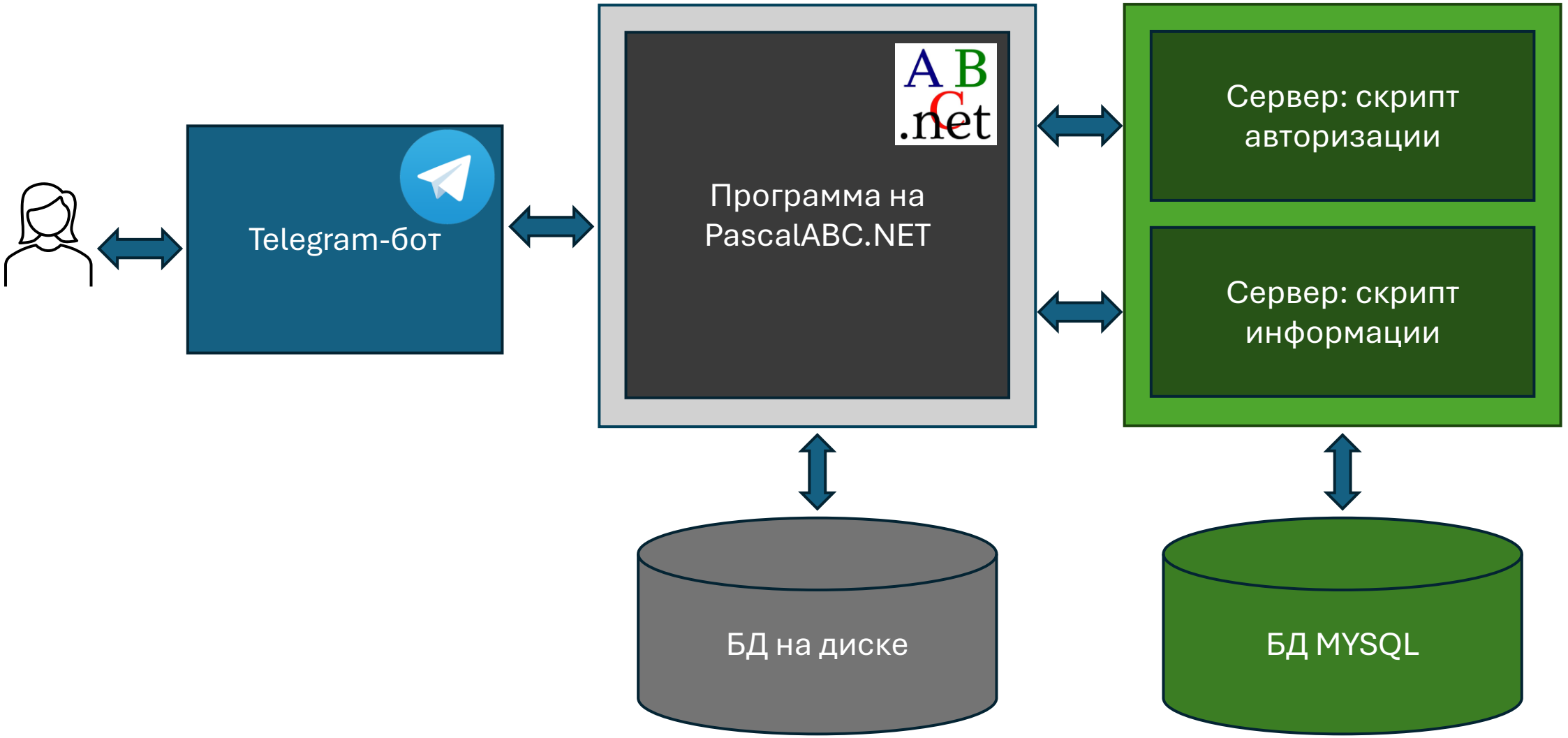
Но в рамках хостинга ЮФУ почта не работает.

То есть как бы работает, но…

1. POP3 и SMTP – классические протоколы.
2. IMAP – зрелый, довольно удобный.
3. OAuth 2.0 (в ЮФУ только так теперь) – получить токен (временный), подключить библиотеки для PHP, выпросить почту, разобраться с Outlook, настроить сервер.

Сторонние сервисы – Яндекс и Mail.ru позволяют работать по POP3 и SMTP, но могут в любой момент заблокировать почтовый ящик.

Проект системы



Как идентифицировать пользователя

Необходимо уметь идентифицировать пользователя, сохранять/изменять/удалять данные. При необходимости восстанавливать пароль.

Telegram – одна из самых удобных систем в этом плане, если использовать идентификатор пользователя (int64).

Недостаток – нужен постоянно работающий компьютер, и аккуратность в написании программы (устойчивость к исключительным ситуациям).

Достоинства – проще поддерживать (по сравнению с серверными скриптами на PHP), проще тестировать.

Выводы

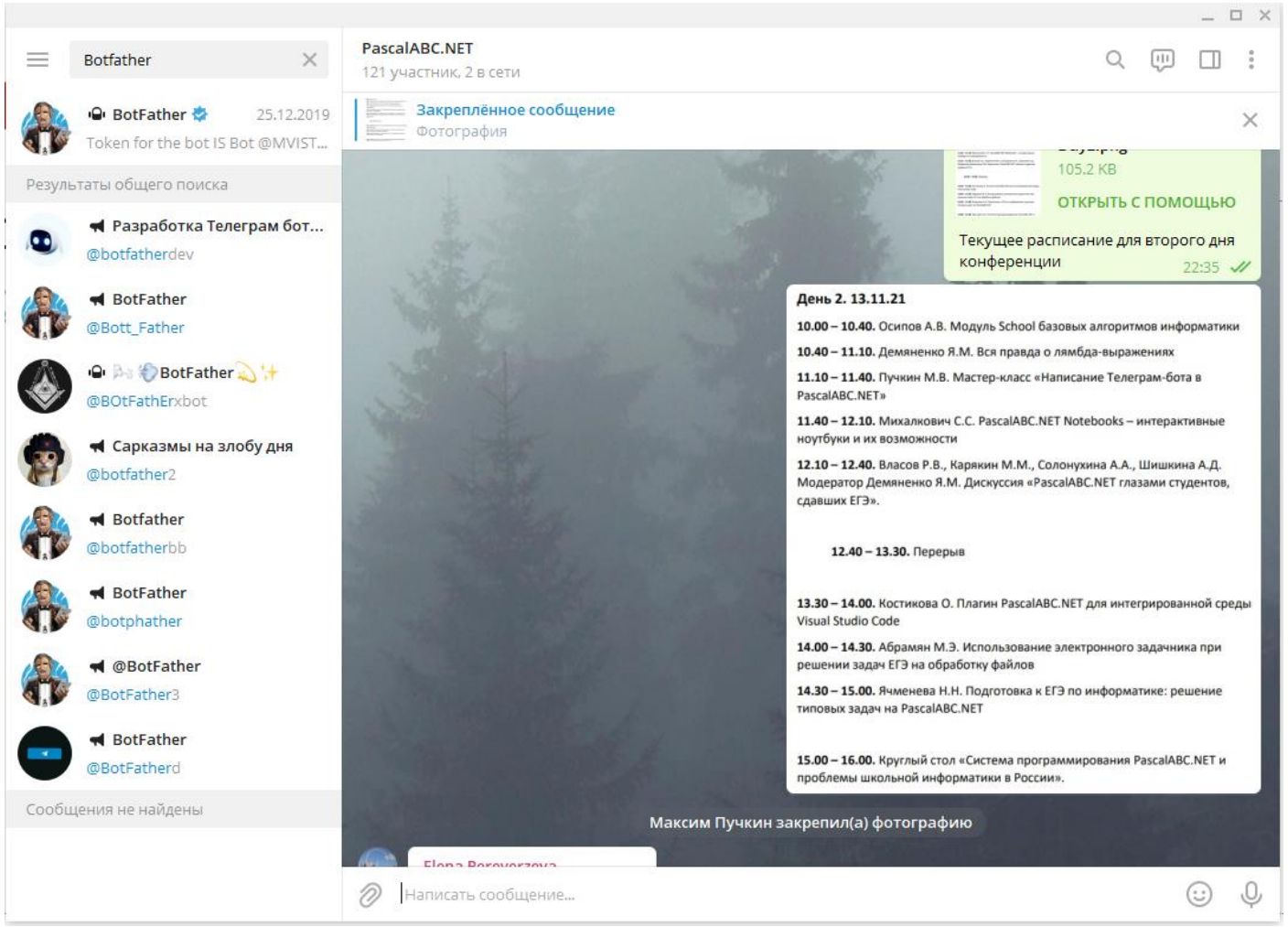
Telegram в качестве платформы в данный момент является идеальным средством коммуникации с пользователями в рамках малых и средних проектов.

Убирает вопросы безопасности, проблемы с отправкой почты (спам/фильтры/время отклика), устойчив и распространён.

Средство общения с уникальными пользователями идеальное (в рамках онлайн-парадигмы).

Как создать бота

1. Открыть чат BotFather
2. Нажать кнопку /
3. Выбрать /newbot
4. Придумать имя бота
5. Если всё ОК – ID бота есть
6. Добавить описание
7. Добавить about
8. Придумать список команд

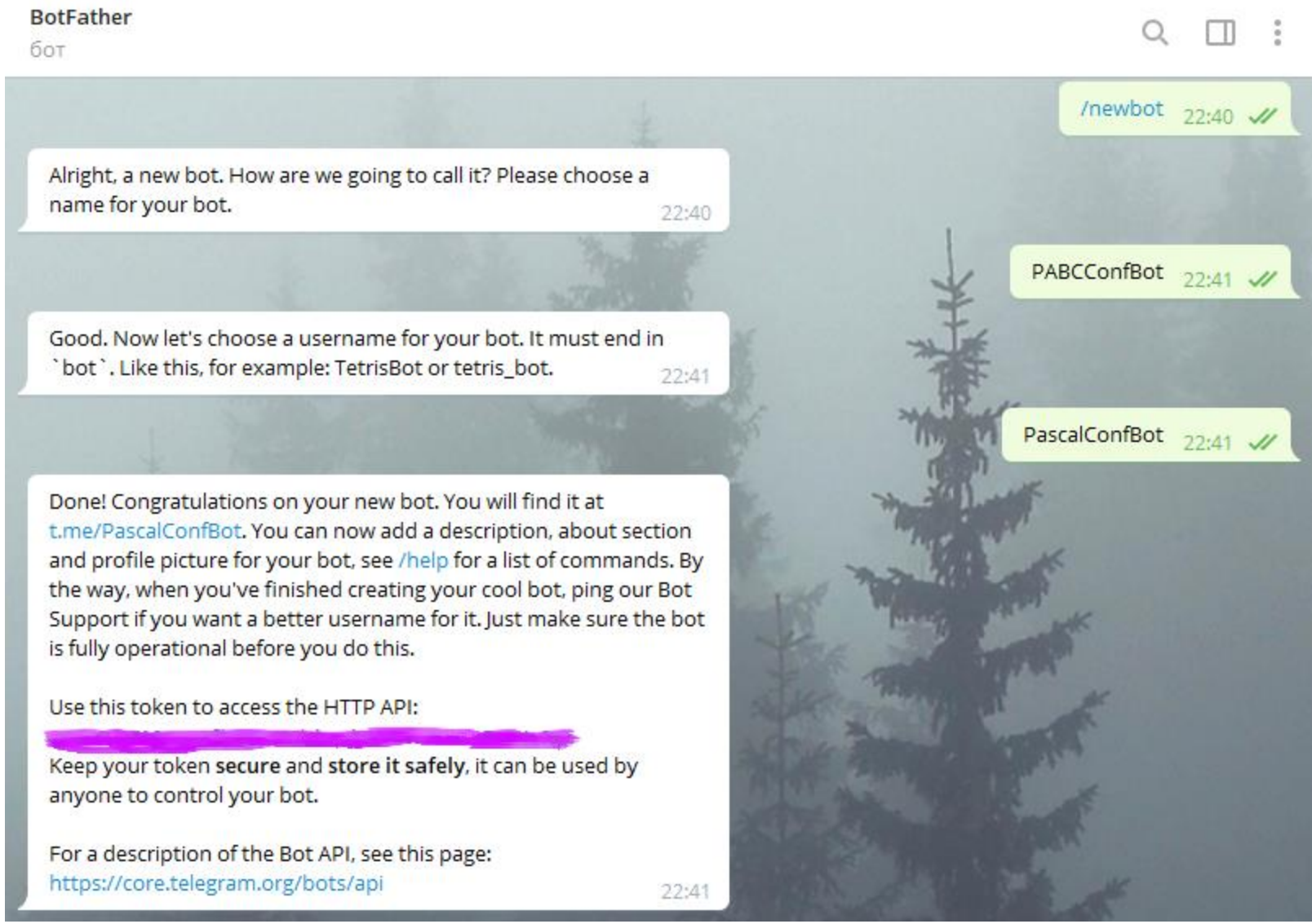


Настройка бота

Общение – как с собеседником.

Если у вас уже есть боты, то система будет переспрашивать имя (кнопкой) каждый раз.

Токен бота никому не показывайте – это, по сути, пароль, зная который можно общаться от имени бота. Можно идентификатор отозвать, но лучше не надо.



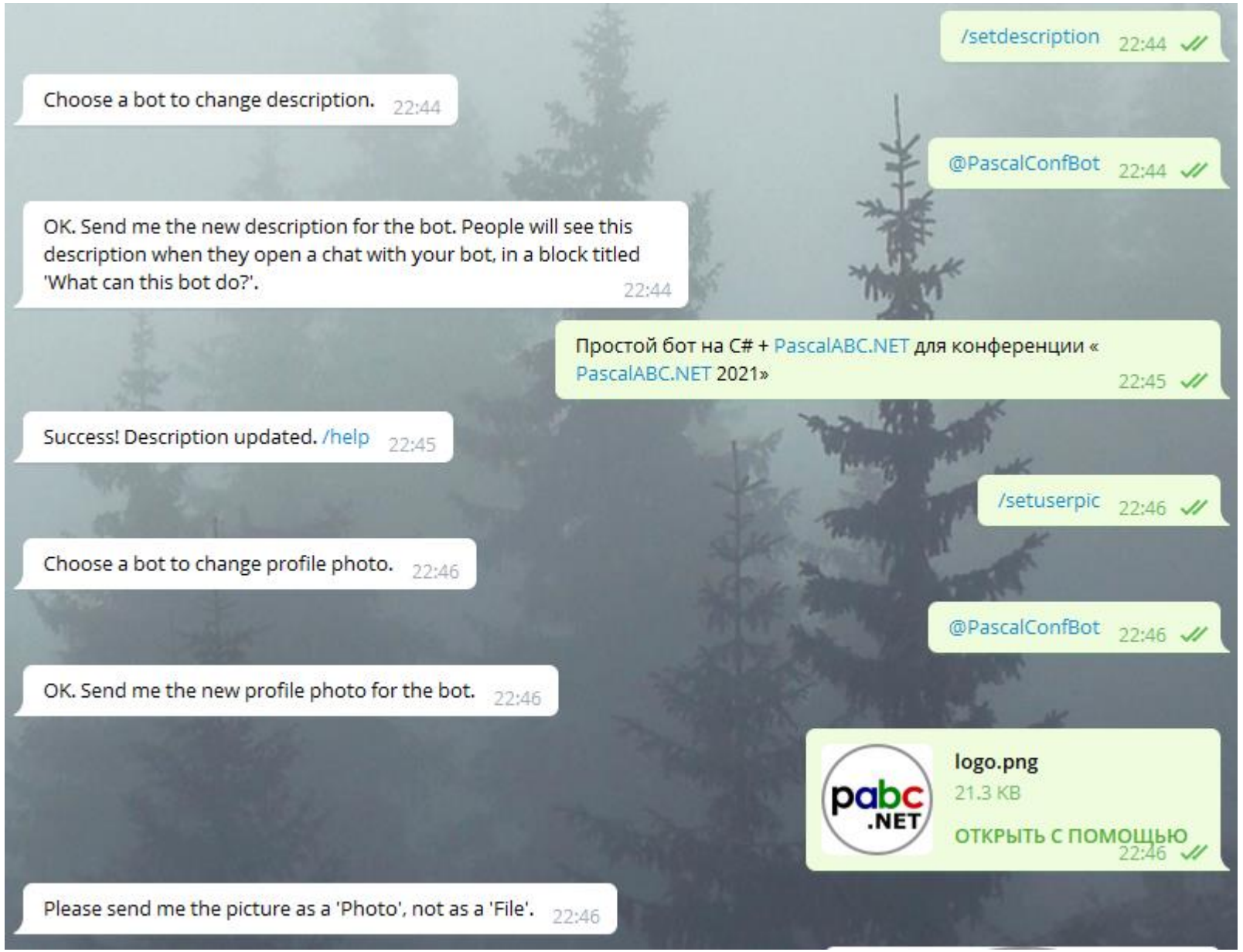
Настройка бота

Боты принимают:

1. Строки
2. Команды
3. Кнопки
4. Изображения
5. Звук

По умолчанию нельзя встраивать бота в беседы, только общение напрямую. Также по умолчанию нельзя в группы добавлять.

Меняется параметром inline и groups.



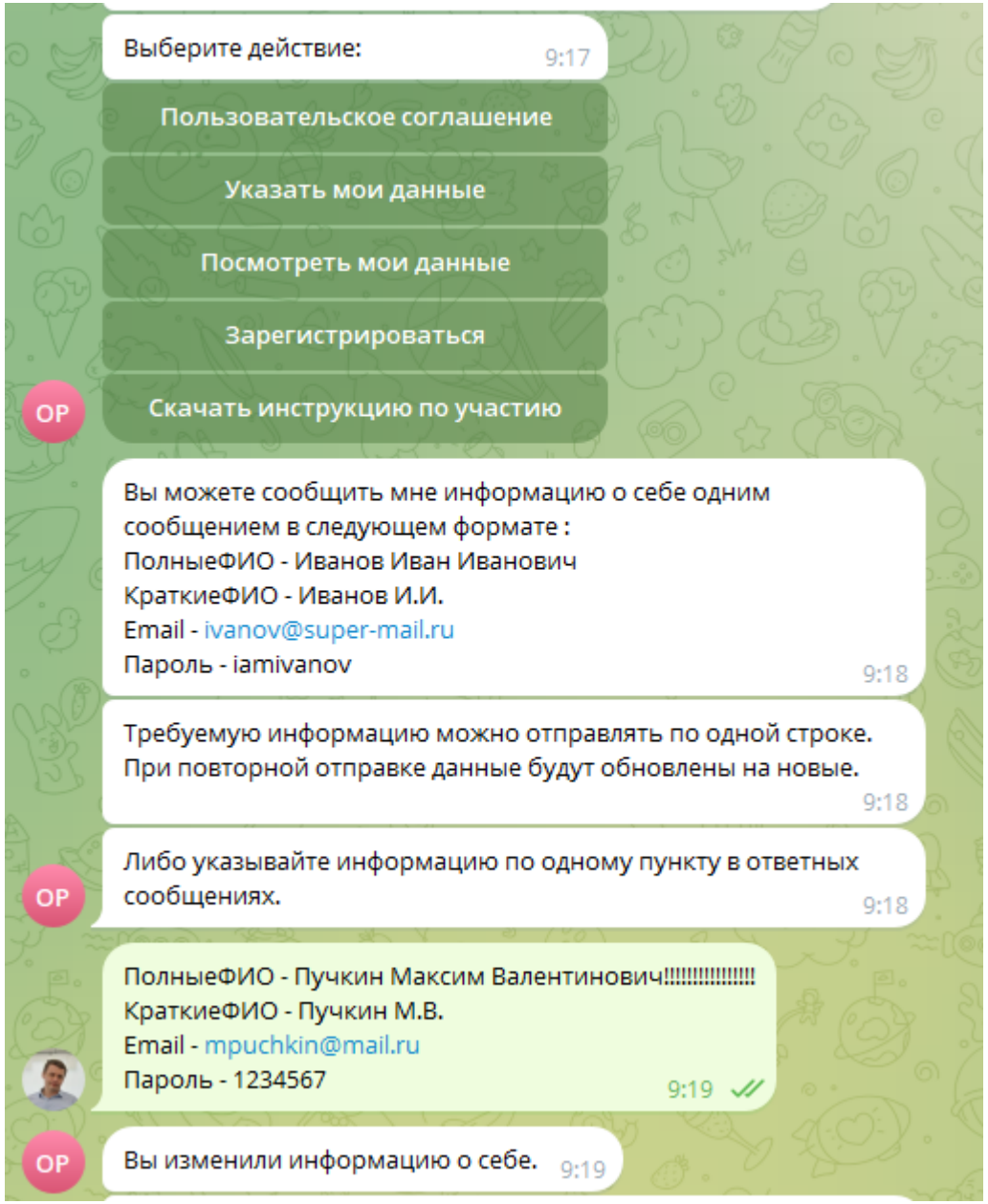
Проблемы

Была попытка сделать процесс регистрации аналогичным тому, который используется при создании бота с помощью BotFather.

Не сработало – большой процент ошибок, связанных как со слабой проработкой логики, так и с отсутствием опыта у пользователей.

Как итог – в базу посыпались вместо ФИО строки с полной информацией.

Тестировать нужно долго, и прорабатывать на пользователях.



Написание библиотеки бота – 2021

- Пишем основу на C# – из-за мелких несоответствий. Методы асинхронные, а поддержки соответствующих механизмов в PascalABC.NET нет. Нужны async/await и всё, что связано с асинхронной работой.
- Доступно большое количество dll-библиотек, желательно смотреть документацию на официальном сайте, и оттуда переходить по ссылкам на соответствующие проекты в GitHub.
- Библиотеки – dll, легко использовать в проектах PascalABC.NET, однако иногда проще написать «обёртку» – библиотеку, использующую библиотеку. Другой вариант – просто подключить управляемую dll в PascalABC.NET и написать свой модуль.
- Нам нужен модуль с классом, который умеет соединяться с ботом, принимать строки текста, и отправлять текст, фото, файлы. С кнопками и командами работать немного сложнее, но в целом это тоже можно добавить. Список команд фиксируется в BotFather.

TelegramBot.API

- Самая распространённая библиотека, однако проблемы с версиями.
- Последняя – 22 – не работает с NET Framework 4.0, нужна 6.0 или выше. «Завелась» версия 16, её хватает.
- Содержит довольно проработанный и обширный набор классов, которые можно использовать «напрямую», однако такой код не соответствует стилистике PascalABC.NET.
- При работе с такими библиотеками теряется весь смысл работы с PascalABC.NET – проще сразу на C# в Visual Studio работать.
- Как переписывать? Класс-обёртку, в котором переписывать основные методы и типы?

Пример кода

```
begin
  // Объект для отправки HTTP-запросов на сервер
  serv := new ServHttpProvider('https://lightpt.sfedu.ru');
  // Объект для работы с базой школьников (можно перенести на серверную часть)
  db := new UsersDatabase('users.txt');

  //  Собственно телеграм-бот
  var botik := new TelegramBot(ReadAllText('Bottoken.txt'));
  //  Обработчик любых сообщений от пользователей
  botik.OnAnyMessage += OnMessage;
  //  Запуск бота
  botik.Act;

  //  Цикл сброса на диск базы пользователей каждые 100 секунд
  while true do
  begin
    sleep(10000);
    db.SaveToFile('users.txt');
  end;
end.
```

Пример плохого кода - 1

```
//  Обработчик любых сообщений
procedure OnMessage(bot : TBot; Args : MsgArgs);
begin
  //  Если такого пользователя нет в базе - добавить
  if not db.ContainsKey(Args.Message.Chat.Id.ToString) then
    db.Add(new UserInfo(Args.Message.Chat.Id.ToString));

  if Args.Message.Text = '/start' then
  begin
    //  Отправляем приветствие и клавиатуру
    SendInlineKeyboard(bot, Args.Message.Chat.Id); exit;
  end;
```

Пример плохого кода - 2

```
/// <summary>
/// Проверка текущей информации о пользователе и регистрация в качестве участника
/// </summary>
procedure RegisterMe(bot : TBot; callbackEvenArgs : CallbackQueryEventArgs);
begin
  ///  Проверить данные, отослать на сервер и вернуть результат
  if not db.ContainsKey(callbackEvenArgs.CallbackQuery.Message.Chat.Id.ToString) then
    begin
      db.Add(new UserInfo(callbackEvenArgs.CallbackQuery.Message.Chat.Id.ToString));
      bot.SendText(callbackEvenArgs.CallbackQuery.Message.Chat.Id, 'Для регистрации
необходимо указать данные о себе!');
      exit;
    end;
```

Пример плохого кода - 3

```
/// <summary>
/// Отправка лицензионного сообщения
/// </summary>
procedure Agreement(bot : TBot; callbackEventArgs :
Telegram.Bot.Args.CallbackQueryEventArgs);
begin
  bot.SendText(callbackEventArgs.CallbackQuery.Message.Chat.Id,
ReadAllText('agreement.txt'));
  SendInlineKeyboard(bot, callbackEventArgs.CallbackQuery.Message.Chat.Id);
end;
```

Переопределение

```
unit TelegramABC;
{$reference Newtonsoft.Json.dll}
{$reference Telegram.Bot.dll}
{$reference TelegramBotLib.dll}

interface
uses TelegramBotLib;

type CallbackQueryEventArgs = Telegram.Bot.Args.CallbackQueryEventArgs;

//  Получение идентификатора чата
function Id(self : CallbackQueryEventArgs) : int64; extensionmethod;
begin
  Result := callbackEventArgs.CallbackQuery.Message.Chat.Id;
end;
```

Примеры кода

```
/// <summary>
/// Отправка лицензионного сообщения
/// </summary>
procedure Agreement(bot : TBot; callbackEventArgs : Telegram.Bot.Args.CallbackQueryEventArgs);
begin
  bot.SendText(callbackEventArgs.CallbackQuery.Message.Chat.Id, ReadAllText('agreement.txt'));
  SendInlineKeyboard(bot, callbackEventArgs.CallbackQuery.Message.Chat.Id);
end;

/// <summary>
/// Отправка лицензионного сообщения
/// </summary>
procedure Agreement(bot : TBot; callbackEventArgs : CallbackQueryEventArgs);
begin
  bot.SendText(callbackEventArgs.Id, ReadAllText('agreement.txt'));
  SendInlineKeyboard(bot, callbackEventArgs.Id);
end;
```

Отправка клавиатуры

```
/// <summary>
/// Отправка клавиатуры с набором кнопок
/// </summary>
procedure SendInlineKeyboard(bot : TBot; id : int64);
begin
    var btns := new List<CallbackButton>();
    btns.Add(new CallbackButton('Пользовательское соглашение', 'agreement', Agreement));
    btns.Add(new CallbackButton('Указать мои данные', 'setdata', SetData));
    btns.Add(new CallbackButton('Посмотреть мои данные', 'viewinfo', ViewMe));
    if AllDataSend(id) = 0 then
      btns.Add(new CallbackButton('Зарегистрироваться', 'registerme', RegisterMe));
    btns.Add(new CallbackButton('Скачать инструкцию по участию', 'instruction',
SendInstruction));
    bot.SendInlineKeyboard(id,'Выберите действие:', btns);
end;
```

Спасибо за внимание!