УДК 004

Создание чат-бота в мессенджере Telegram

Эрдман Александр Алексеевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В статье представлено руководство по созданию чат-бота в популярном мессенджере Telegram. Программирование бота осуществляется на языке программирования Python.
Ключевые слова: python, создание бота, telegram бот

Creating a bot in the Telegram messenger

Erdman Alexander Alekseevich
Sholom-Aleichem Priamursky State University
Student

Abstract
The article presents a guide to creating a chatbot in the popular Telegram messenger. The bot is programmed in the Python programming language.
Keywords: python, creating a bot, telegram bot

1. Введение

1.1 Актуальность

На сегодняшний день невозможно представить социальные сети и мессенджеры без чат-ботов, которые с каждым годом становятся не только популярнее, но даже становятся частью цифровой экосистемы мессенджеров и социальных сетей. Необходимость в чат-ботах возникла ещё в конце прошлого века, тогда боты создавались в качестве развлечения пользователей. Примером может служить чат-бот «Телевикторина», написанный на базе протокола IRC. Его функция заключалась в том, чтобы задавать различного рода вопросы, на которые пользователи должны были отвечать, а бот в свою очередь должен был либо принять ответ, либо отклонить. С развитием данной индустрии начали появляться боты, которые выполняли роль консультанта. Сейчас практически в каждой группе того или иного мессенджера можно встретить бота, который сможет ответить на все основные вопросы пользователя. Такие боты существенно облегчают труд консультантов той или иной сферы деятельности. Востребованность чат-ботов постоянно растёт, а вместе с этим растёт спрос на грамотных специалистов в данной сфере. Поэтому сегодня актуально уметь создавать своего чат-бота, ведь благодаря ему можно заработать приличные деньги за относительно небольшой и несложный труд.

1.2 Обзор исследований

А.В. Иванова и А.А. Кузьменко создали чат-бота на основе нейронной сети, которая в свою очередь подразумевает использование различных методов и способов анализа фраз и предложений [1]. Т.А. Шестаков разработал чат-бота для автоматизации работы техподдержки [2]. С.С. Гречихин оценил эффективность современных мессенджеров и чат-ботов в программе дистанционного обучения [3]. Э.Р. Халимова, И.Ю. Карякин, Л.Н. Бакановская и О.С. Вунш создали чат-бота на базе программы для расчёта вероятности заболевания сахарным диабетом [4]. Н.Д. Синева, А.В. Хижная, А.А. Мазунова, А.Н. Сидрова рассмотрели чат-бота в качестве современной технологии образования [5]. С.Ю. Смирнов рассмотрел чат-ботов, как самую прогрессивную и перспективную технологию в мире интернета [6].

1.3 Цель исследования

Целью исследования является создание простого чат-бота типа «эхо-бот» на языке программирования Python, а также усложнение его функций. Чат-бот будет способен приветствовать пользователя и отвечать на простейшие вопросы.

2 Материалы и методы

Для создания чат-бота используется библиотека pyTelegramBotAPI, языке программирования Python, Telegram Desktop. В качестве IDE используется PyCharm.

3 Результаты и обсуждения

В данном исследование будет рассмотрено создание чат-бота типа «Эхо-бот», так как он является довольно простым для понимания и написания. Для этого понадобиться 2 файла «.py» – это config.py и bot.py. Config.py отвечает за хранение и дальнейшую ссылку на токен, bot.py – это непосредственно сам бот. Перейдём к созданию. Первым шагом является установка библиотеки работы с Telegram API. Для этого необходимо перейти в terminal PyCharm, прописать команду «pip install pytelegrambotapi» и дождаться установки. Дальше нужно открыть файл config.py и прописать токен бота. Помимо токена, в данном файле также можно прописывать настройки бота. Для получения токена бота необходимо зайти в Telegram Desktop, в поиске чатов написать «BotFather». Данный бот является прародителем всех ботов в мессенджере Telegram, через него можно получить токен для своего бота, а также произвести первичную настройку (рис. 1). После нужно начать диалог с ботом командой /start. Дальше необходимо прописать команду /newbot (команда на создание бота). После BotFather предложит выбрать имя и никнейм бота. Заметьте, что имя может использоваться любое, а никнейм должен быть уникальным, причём никнейм обязательно должен оканчиваться на «bot». Для примера имя бота «bot1», никнейм бота «uniaue_niknamebot». После ввода никнейма бота будет создан токен, который нужно скопировать и прописать в config.py (рис. 2).

Дополнительно можно добавить картинку боту. Для этого необходимо прописать команду /setuserpic, далее выбрать какому боту присвоить картинку (в данном случае будет один единственный) и непосредственно выбрать картинку. После того, как токен был прописан, нужно открыть bot.py и написать код самого эхо-бота (рис. 3). Для проверки на наличие ошибок можно запустить код и убедиться, что ошибок нет. Дальше для проверки работоспособности самого бота нужно перейти в Telegram и написать боту что-нибудь. Нужно заметить, что при входе в чат с ботом, всегда отправляется первая команда /start. Так как это эхо-бот, то он должен отражать наши сообщения (рис. 4). После проверки работоспособности, можно усложнить функционал бота. Для примера, можно добавить возможность боту приветствовать пользователя причём не только текстом, но и стикером, а также клавиатуру из двух кнопок «Случайное число» и «Как дела?». Чтобы бот мог отправлять стикер во время приветствия, его нужно скачать. Для Telegram используется стикеры формата «name_sticker.tgs». Данные стикеры можно скачать прямо с телеграмм, для этого достаточно отправить стикер либо самому себе, либо кому-нибудь. Дальше щёлкнуть правой кнопкой мыши по стикеру и выбрать «Save as..» и сохранить в папку «stick», которая находится в директории файлов bot.py и config.py. После сохранения стикера может приступить к реализации приветствия бота. Для этого необходимо добавить новую функцию, которая будет отвечать за обработку команды /start. Благодаря этой функции, пользователь, запустивший бота впервые, получит текст с приветствием, а также приветственный стикер. Нужно обратить внимание на метод отправки стикера. Данный метод схож с методом отправки сообщения, только в качестве сообщения ему нужно передать дескриптор открытого файла стикера. Формирование сообщений Telegram API осуществляется двумя вариантами разметки – Markdown и HTML. В данном случае будет использована разметка сообщения при помощи HTML – parse_mode=’html’. Код приветствия готов (рис. 5). После нужно проверить работоспособность новой функции бота (рис. 6). Далее произведём ещё больше усложнение функций. Преобразуем примитивного эхо-бота в более продвинутого чат-бота. Для этого нужно добавить клавиатуру бота, через которую с ним можно непосредственно вести общение. Существует два типа клавиатур бота: ReplyKeyboardMarkup, которая отображается внизу под полем ввода сообщения и InlineKeyboardMarkup, которая прикрепляется к сообщениям. Для начала будет создана ReplyKeyboardMarkup. Она будет иметь 2 кнопки: «Как дела?» и «Случайное число». Нужно импортировать две библиотеки: types (from telebot) и random. Первая библиотека отвечает за клавиатуры, а вторая отвечает за действие кнопки «Случайное число». Код клавиатуры выглядит следующим образом (рис. 7). После написания клавиатуры нужно удалить функцию возврата сообщений пользователя (эхо функцию) и вместо неё прописать реакцию бота на нажатие данных кнопок (рис. 8). Проверка работоспособности новых функций бота (рис. 9).

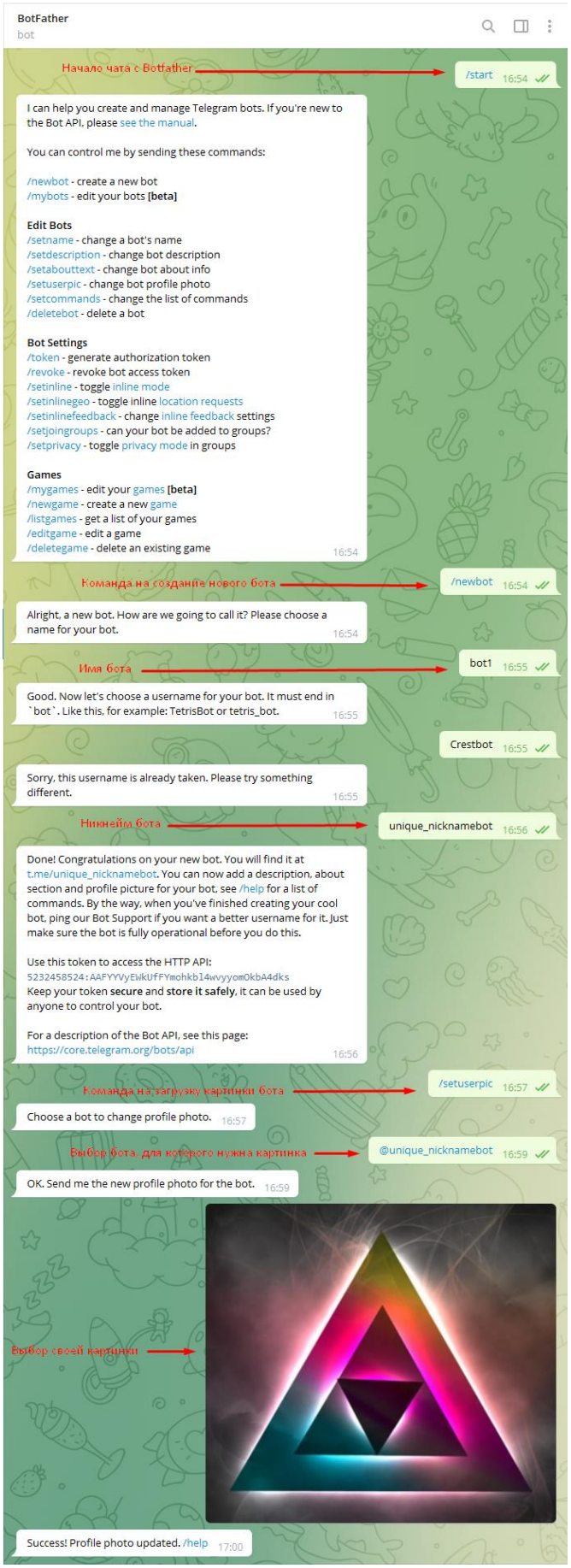

Рис. 1. Получение токена и первичная настройка бота

```
Bot.py  config.py
TOKEN = '5232458524:AAFYYVyEWkUfFYmohkbl4wvyyomOkbA4dks' # bot token from @BotFather
```

Рис. 2. Добавление токена в config.py

```python
import telebot
import config

bot = telebot.TeleBot(config.TOKEN)

@bot.message_handler(content_types=['text'])
def f1(message):
    bot.send_message(message.chat.id, message.text)
# Run
bot.polling(none_stop=True)
```

Рис. 3. Код эхо-бота

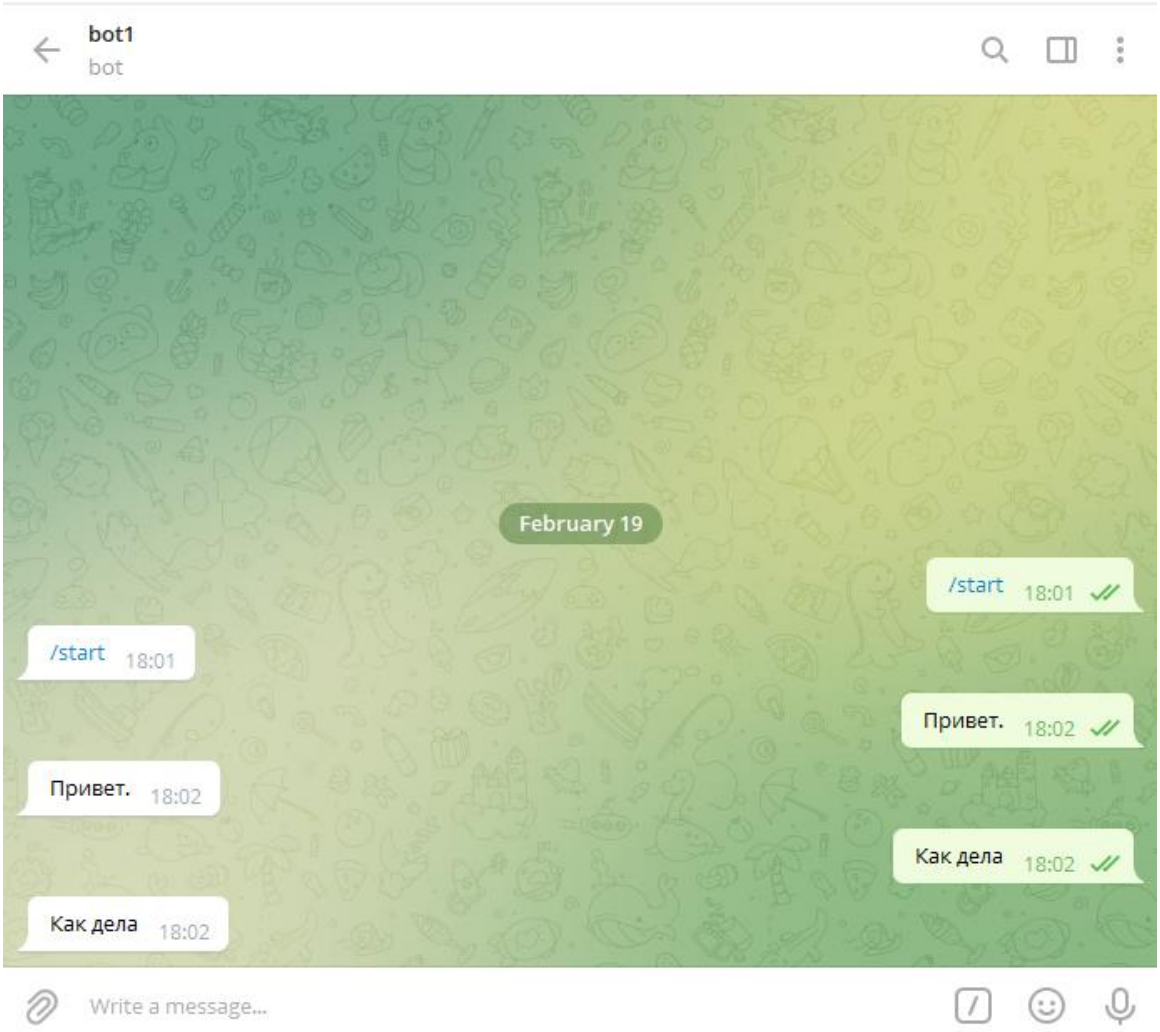

Рис. 4. Проверка эхо-бота

```python
import telebot
import config

bot = telebot.TeleBot(config.TOKEN)
@bot.message_handler(commands=['start'])
def welcome(message):
    sti = open('stick/hi.tgs', 'rb')
    bot.send_sticker(message.chat.id, sti)

    bot.send_message(message.chat.id, "Hello, my friend, {0.first_name}!\nЯ - <b>{1.first_name}</b>, тестовый бот.".format(message.from_user, bot.get_me()),
        parse_mode='html')
# Run
bot.polling(none_stop=True)
```

Рис. 5. Код функции приветсвия чат-бота

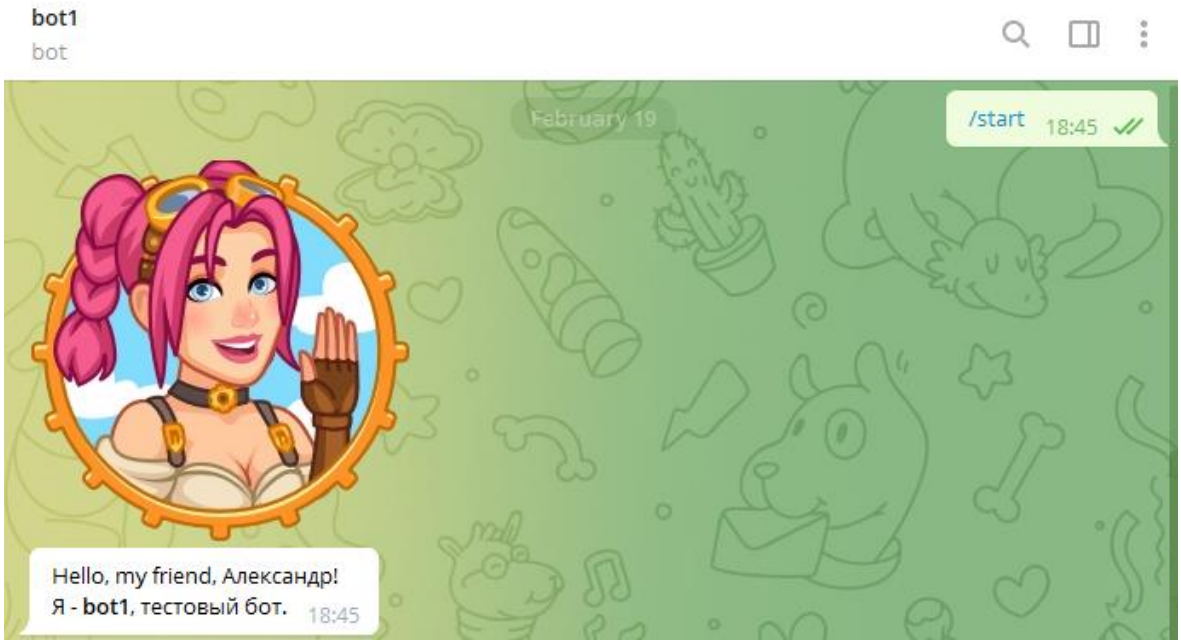

Рис. 6. Проверка работоспособности функции приветствия бота

```python
def welcome(message):
    sti = open('stick/hi.tgs', 'rb')
    bot.send_sticker(message.chat.id, sti)
    # keyboard
    markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
    item1 = types.KeyboardButton("Случайное число")
    item2 = types.KeyboardButton("Как дела?")
    markup.add(item1, item2)

    bot.send_message(message.chat.id, "Hello, my friend, {0.first_name}!\nЯ - <b>{1.first_name}</b>, тестовый бот.".format(message.from_user, bot.get_me()),
        parse_mode='html', reply_markup=markup)
```

Рис. 7. Код клавиатуры типа ReplyKeyboardMarkup

```python
@bot.message_handler(content_types=['text'])
def f1(message):
    if message.chat.type == 'private':
        if message.text == 'Случайное число':
            bot.send_message(message.chat.id, str(random.randint(0,100)))
        elif message.text == 'Как дела?':
            bot.send_message(message.chat.id, 'Замечательно! А твои как ?')
        else:
            bot.send_message(message.chat.id, 'Я даже и не знаю что на это ответить :(')
```

Рис. 8. Код реакции бота на кнопки клавиатуры

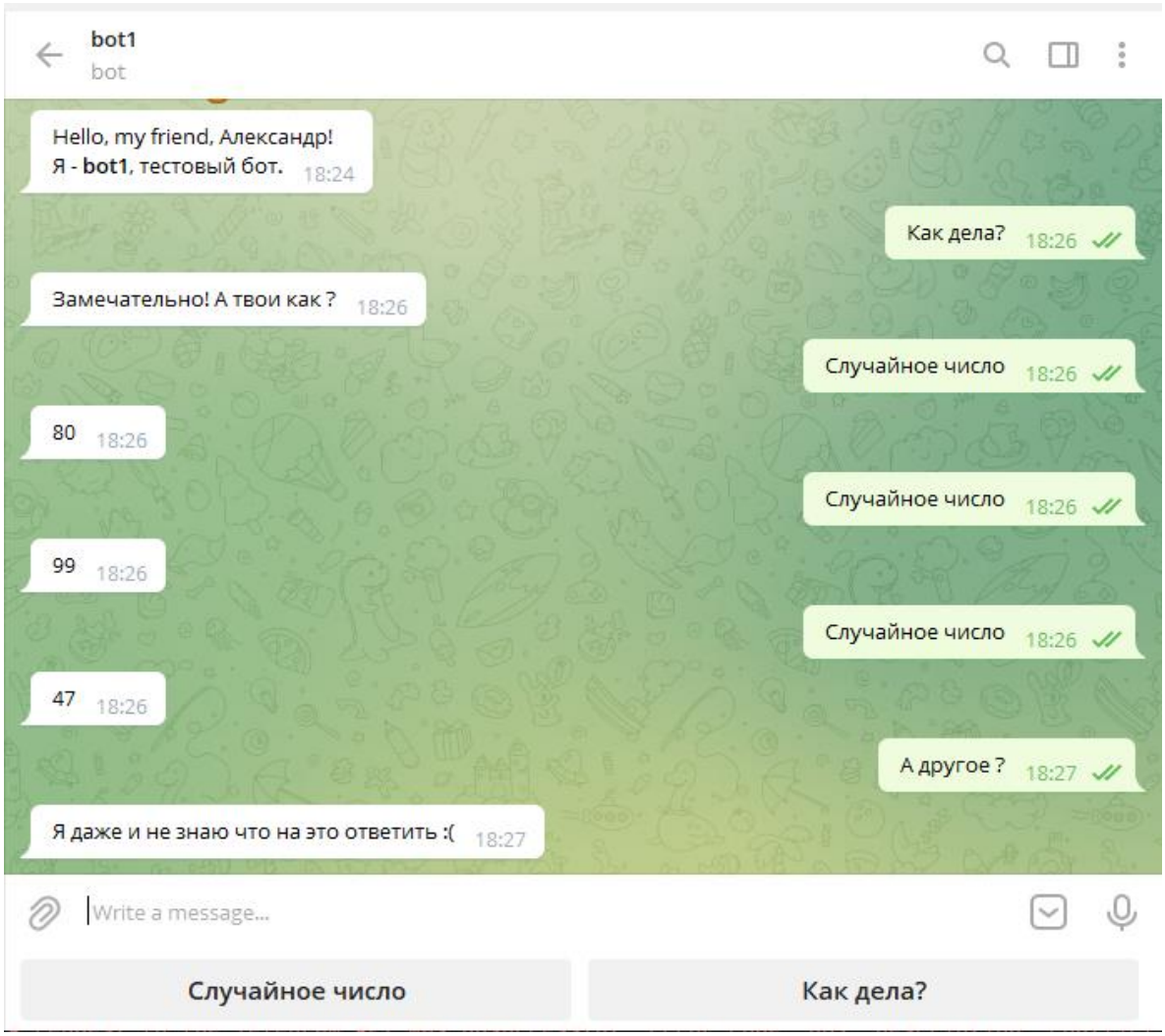

Рис. 9. Проверка работы клавиатуры и реакции бота на кнопки

После удачной проверки работоспособности можно приступать к созданию клавиатуры типа InlineKeyboardMarkup (рис. 10.)

```
@bot.message_handler(content_types=['text'])
def f1(message):
    if message.chat.type == 'private':
        if message.text == 'Случайное число':
            bot.send_message(message.chat.id, str(random.randint(0,100)))
        elif message.text == 'Как дела?':
            markup = types.InlineKeyboardMarkup(row_width=2)
            item1 = types.InlineKeyboardButton("Хорошо", callback_data='good')
            item2 = types.InlineKeyboardButton("Могло быть и лучше", callback_data='bad')
            markup.add(item1, item2)
            bot.send_message(message.chat.id, 'Замечательно! А твои как ?', reply_markup=markup)
        else:
            bot.send_message(message.chat.id, 'Я даже и не знаю что на это ответить :(')
```

Рис. 10. Создание клавиатуры типа InlineKeyboardMarkup

Перед началом проверки работоспособности клавиатуры нужно прописать функцию обработки InlineKeyboardMarkup клавиатуры. Осуществляется данная функция по средствам отдельного метода и отдельного декоратора. Здесь прописываются условия, а также отправка того

или иного сообщения бота в зависимости от того, какую кнопку нажмёт пользователь (рис. 11).

```python
@bot.callback_query_handler(func=lambda call: True)
def callback_inline(call):
    try:
        if call.message:
            if call.data == 'good':
                bot.send_message(call.message.chat.id, 'Я очень рад за тебя)')
            elif call.data == 'bad':
                bot.send_message(call.message.chat.id, 'Эх, сочувствую')
    except Exception as e:
        print(repr(e))
```

Рис. 11. Функция обработки InlineKeyboardMarkup клавиатуры

На этом этапе усложнение функционала бота остановимся. Так как чат-бот завершён можно произвести полную проверку бота (рис. 12). Полный код чат-бота (рис. 13).

Рис. 12. Полная проверка функциональности и работоспособности чат-бота

```python
import telebot
import config
import random

from telebot import types

bot = telebot.TeleBot(config.TOKEN)
@bot.message_handler(commands=['start'])
def welcome(message):
    sti = open('stick/hi.tgs', 'rb')
    bot.send_sticker(message.chat.id, sti)
    # keyboard
    markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
    item1 = types.KeyboardButton("Случайное число")
    item2 = types.KeyboardButton("Как дела?")
    markup.add(item1, item2)

    bot.send_message(message.chat.id, "Hello, my friend, {0.first_name}!\nЯ - <b>{1.first_name}</b>, тестовый бот.".format(message.from_user, bot.get_me()),
        parse_mode='html', reply_markup=markup)
@bot.message_handler(content_types=['text'])
def f1(message):
    if message.chat.type == 'private':
        if message.text == 'Случайное число':
            bot.send_message(message.chat.id, str(random.randint(0,100)))
        elif message.text == 'Как дела?':
            markup = types.InlineKeyboardMarkup(row_width=2)
            item1 = types.InlineKeyboardButton("Хорошо", callback_data='good')
            item2 = types.InlineKeyboardButton("Могло быть и лучше", callback_data='bad')
            markup.add(item1, item2)
            bot.send_message(message.chat.id, 'Замечательно! А твои как ?', reply_markup=markup)
        else:
            bot.send_message(message.chat.id, 'Я даже и не знаю что на это ответить :(')

@bot.callback_query_handler(func=lambda call: True)
def callback_inline(call):
    try:
        if call.message:
            if call.data == 'good':
                bot.send_message(call.message.chat.id, 'Я очень рад за тебя)')
            elif call.data == 'bad':
                bot.send_message(call.message.chat.id, 'Эх, сочувствую')
    except Exception as e:
        print(repr(e))
# Run
bot.polling(none_stop=True)
```

Рис. 13. Полный код чат-бота

В результате исследования удалось создать сначала эхо-бота, а после доработки и усложнений функционала преобразовать его в чат-бота.

Библиографический список

1. Иванова А.В., Кузьменко А.А., Филиппов Р.А., Филиппова Л.Б., Сазонова А.С., Леонов Ю.А. Исследование методов обработки текстовой информации и обзор этапов создания модели искусственного интеллекта при создании чат-ботов // Автоматизация и моделирование в проектировании и управлении. 2021. № 2 (12). С. 19-23.
2. Шестаков Т.А. Создание чат-бота для автоматизации работы техподдержки // В сборнике: Новые горизонты. Материалы VII научно-практической конференции с международным участием. 2020. С. 469-471.
3. Гречихин С.С. Дистанционное обучение с помощью образовательных чат-ботов в современных мессенджерах // Балтийский гуманитарный журнал. 2020. Т. 9. № 3 (32). С. 66-68.
4. Халимова Э.Р., Карякин И.Ю., Бакановская Л.Н., Вунш О.С. Чат-бот мессенджера Telegram ("medicinebot") // Свидетельство о регистрации программы для ЭВМ RU 2018663280, 24.10.2018. Заявка № 2018660323 от 25.09.2018.

5. Синева Н.Л., Хижная А.В., Мазунова А.А., Сидоров А.Н. Особенности обучения персонала современной организации посредством чат-ботов // Наука Красноярья. 2020. Т. 9. № 2-3. С. 176-180.
6. Смирнов С.Ю. Чат-боты: настоящее и будущее искусственного интеллекта // Инженерные кадры - будущее инновационной экономики России. 2019. № 8. С. 137-140.