**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет ИУ

Кафедра ИУ5

Курс «Основы информатики»

Отчет по лабораторной работе №6

“Телеграмм бот”

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы 36б:  Соболев С.С. |  | преподаватель каф. \_\_\_\_ |
|  |  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2024г.

**Задача:**

1. Разработайте простого бота для Telegram. Бот должен использовать функциональность создания кнопок.

**Код программы:**

import requests

from telegram import Update, InlineKeyboardButton, InlineKeyboardMarkup

from telegram.ext import ApplicationBuilder, CommandHandler, MessageHandler, CallbackQueryHandler, filters, ContextTypes

API\_KEY = "04ec50d0b755da6e58ae731c"

BASE\_URL = f"https://v6.exchangerate-api.com/v6/{API\_KEY}/latest/"

def get\_exchange\_rate(base\_currency, target\_currency):

    try:

        response = requests.get(BASE\_URL + base\_currency)

        data = response.json()

        if response.status\_code == 200 and "conversion\_rates" in data:

            return data["conversion\_rates"].get(target\_currency)

        else:

            return None

    except Exception as e:

        return None

async def start(update: Update, context: ContextTypes.DEFAULT\_TYPE):

    context.user\_data.clear()

    keyboard = [

        [InlineKeyboardButton("Русский", callback\_data="LANG\_RU"),

         InlineKeyboardButton("English", callback\_data="LANG\_EN")]

    ]

    reply\_markup = InlineKeyboardMarkup(keyboard)

    await update.message.reply\_text(

        "Choose your language / Выберите язык:",

        reply\_markup=reply\_markup

    )

async def set\_language(update: Update, context: ContextTypes.DEFAULT\_TYPE):

    query = update.callback\_query

    await query.answer()

    if query.data == "LANG\_RU":

        context.user\_data["language"] = "RU"

        await query.edit\_message\_text("Вы выбрали русский язык.")

        await show\_currency\_menu(update, context, lang="RU")

    elif query.data == "LANG\_EN":

        context.user\_data["language"] = "EN"

        await query.edit\_message\_text("You have selected English.")

        await show\_currency\_menu(update, context, lang="EN")

async def show\_currency\_menu(update, context, lang):

    keyboard = [

        [InlineKeyboardButton("USD -> EUR", callback\_data="USD\_EUR"),

         InlineKeyboardButton("EUR -> USD", callback\_data="EUR\_USD")]

    ]

    reply\_markup = InlineKeyboardMarkup(keyboard)

    if lang == "RU":

        message = "Выберите направление перевода:"

    else:

        message = "Select the conversion direction:"

    if update.callback\_query:

        await update.callback\_query.edit\_message\_text(

            message,

            reply\_markup=reply\_markup

        )

    else:

        await update.message.reply\_text(

            message,

            reply\_markup=reply\_markup

        )

async def handle\_currency\_selection(update: Update, context: ContextTypes.DEFAULT\_TYPE):

    query = update.callback\_query

    await query.answer()

    if query.data == "USD\_EUR":

        context.user\_data["conversion"] = ("USD", "EUR")

        lang = context.user\_data.get("language", "EN")

        if lang == "RU":

            await query.edit\_message\_text("Вы выбрали перевод USD -> EUR. Введите сумму для перевода:")

        else:

            await query.edit\_message\_text("You selected USD -> EUR. Enter the amount to convert:")

    elif query.data == "EUR\_USD":

        context.user\_data["conversion"] = ("EUR", "USD")

        lang = context.user\_data.get("language", "EN")

        if lang == "RU":

            await query.edit\_message\_text("Вы выбрали перевод EUR -> USD. Введите сумму для перевода:")

        else:

            await query.edit\_message\_text("You selected EUR -> USD. Enter the amount to convert:")

async def handle\_amount\_input(update: Update, context: ContextTypes.DEFAULT\_TYPE):

    if "conversion" not in context.user\_data:

        lang = context.user\_data.get("language", "EN")

        if lang == "RU":

            await update.message.reply\_text("Сначала выберите направление перевода с помощью команды /start.")

        else:

            await update.message.reply\_text("Please select a conversion direction first using the /start command.")

        return

    try:

        amount = float(update.message.text)

        base\_currency, target\_currency = context.user\_data["conversion"]

        rate = get\_exchange\_rate(base\_currency, target\_currency)

        if rate:

            converted\_amount = amount \* rate

            lang = context.user\_data.get("language", "EN")

            if lang == "RU":

                await update.message.reply\_text(

                    f"{amount} {base\_currency} = {converted\_amount:.2f} {target\_currency}"

                )

            else:

                await update.message.reply\_text(

                    f"{amount} {base\_currency} = {converted\_amount:.2f} {target\_currency}"

                )

        else:

            lang = context.user\_data.get("language", "EN")

            if lang == "RU":

                await update.message.reply\_text("Не удалось получить курс валют. Попробуйте снова позже.")

            else:

                await update.message.reply\_text("Failed to fetch the exchange rate. Please try again later.")

        lang = context.user\_data.get("language", "EN")

        await show\_currency\_menu(update, context, lang=lang)

    except ValueError:

        lang = context.user\_data.get("language", "EN")

        if lang == "RU":

            await update.message.reply\_text("Пожалуйста, введите корректное число.")

        else:

            await update.message.reply\_text("Please enter a valid number.")

if \_\_name\_\_ == "\_\_main\_\_":

    app = ApplicationBuilder().token("8032079486:AAFynMazqWEonqUrAL76tmtMAmJPRlfLG4Y").build()

    app.add\_handler(CommandHandler("start", start))

    app.add\_handler(CallbackQueryHandler(set\_language, pattern="^LANG\_.\*"))

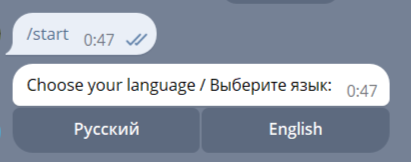
    app.add\_handler(CallbackQueryHandler(handle\_currency\_selection, pattern="^(USD\_EUR|EUR\_USD)$"))

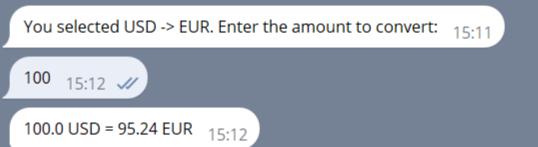
    app.add\_handler(MessageHandler(filters.TEXT & ~filters.COMMAND, handle\_amount\_input))

    print("Бот запущен...")

    app.run\_polling()

**Результат выполнения:**

****

****