**Теоретические материалы к занятию 1**

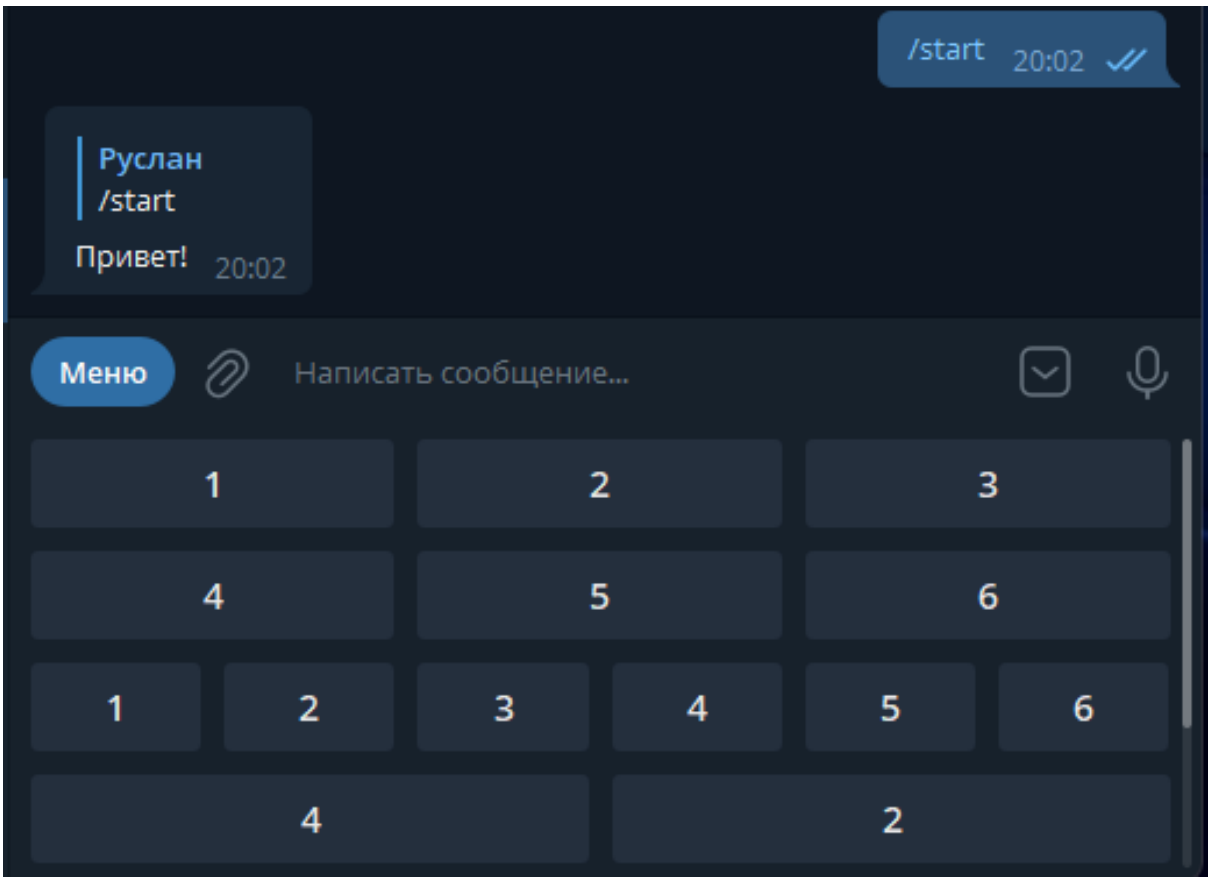
Рассмотрим подробнее метод для составления клавиатур - insert. Он похож на метод add, но начинает добавлять кнопки не с нового ряда, а сначала проверяет, заполнен ли до конца последний ряд. И если нет, то сначала добавляет кнопки туда, а переносит строку только при достижении указанного лимита.

```
kb5 = ReplyKeyboardMarkup()

kb5.add(
   button1, button2, button3, button4, button5, button6
)
kb5.row(
   button1, button2, button3, button4, button5, button6
)

kb5.row(button4, button2)
kb5.add(button3, button2)
kb5.insert(button1)
kb5.insert(button6)
kb5.insert(KeyboardButton('9'))
```

```
@dp.message_handler(commands=['start'])
async def process_start_command(message: types.Message):
   await message.reply("Привет!", reply_markup=kb9)
```




После каждой отправки ботом пользователю клавиатуры, последняя заменяет предыдущую. Поэтому пользователь всегда может открыть её, даже когда по контексту она не нужна. Для того, чтобы у пользователя в клиенте клавиатура убралась совсем, нужно отправить ему ReplyKeboardRemove:

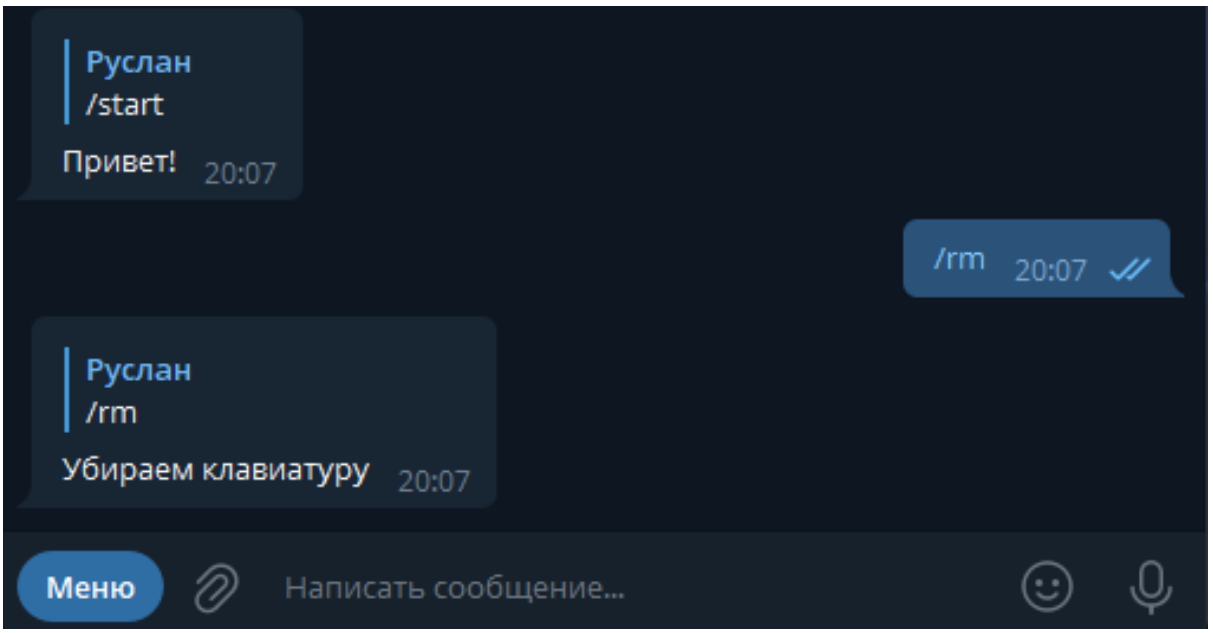
*keyboards.py*

```
from aiogram.types import ReplyKeyboardMarkup, KeyboardButton,
ReplyKeyboardRemove
```

*bot.py*

```
from keyboards import kb5, ReplyKeyboardRemove

@dp.message_handler(commands=['rm'])
async def process_rm_command(message: types.Message):
    await message.reply("Убираем клавиатуру",
reply_markup=ReplyKeyboardRemove())
```




Теперь перейдем к инлайн клавиатурам. Они имеют больше параметров, поэтому позволяют нам делать больше разных вещей. Самое популярное использование — как кнопка, являющаяся *шорткатом* для какого-то действия. То есть «если пользователь нажал кнопку X, сделать Y». И под Y можно понимать вообще что угодно, так как это уже не ограничивается даже API. Рассмотрим наглядно, для этого передадим в инициализатор значение `callback_data`:

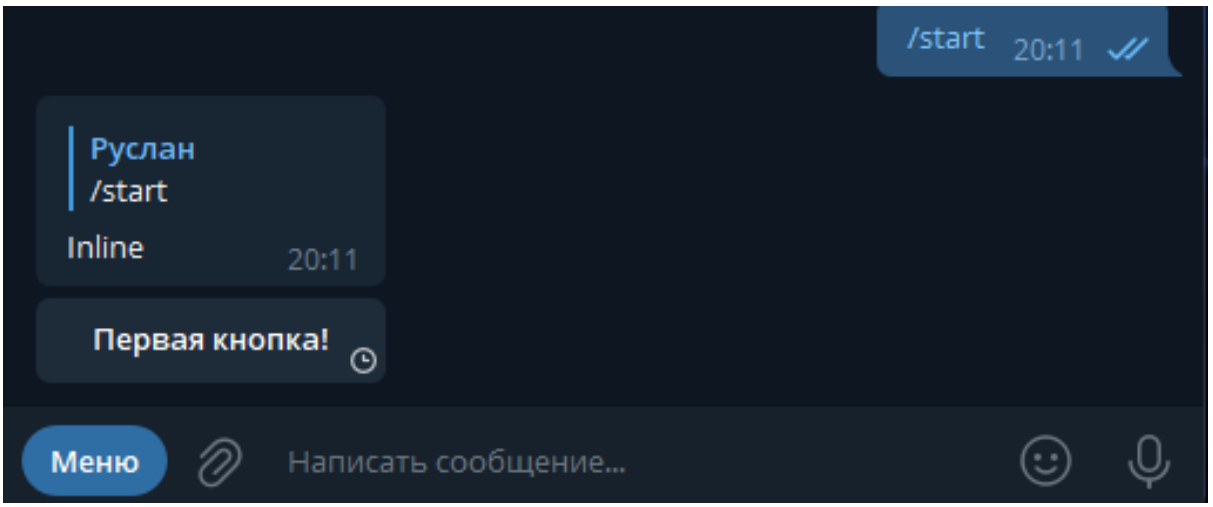
*keyboards.py*

```
from aiogram.types import  InlineKeyboardButton,
InlineKeyboardMarkup


inline_btn_1 = InlineKeyboardButton('Первая кнопка!',
callback_data='button1')
inline_kb1 = InlineKeyboardMarkup().add(inline_btn_1)
```

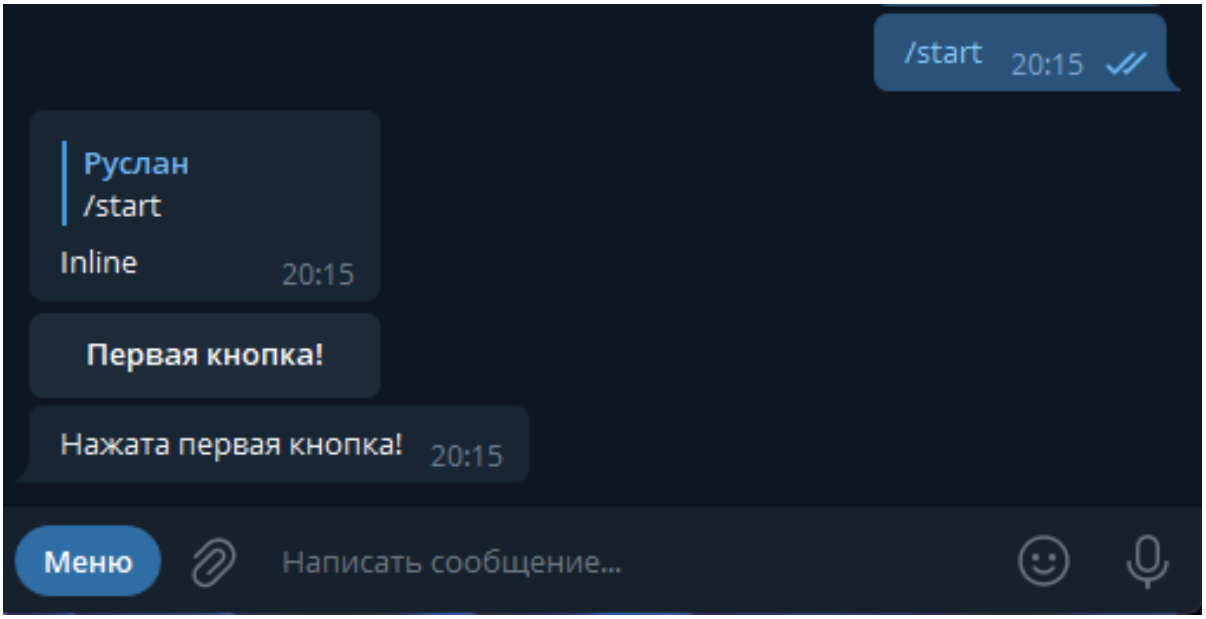
*bot.py*

```
@dp.message_handler(commands=['start'])
async def process_rm_command(message: types.Message):
    await message.reply("Inline", reply_markup=inline_kb1)
```




Нажимаем кнопку и.. ничего не происходит! Почему? Если у бота было включено логирование, то вы могли заметить, что приходит обновление типа CallbackQuery. Так вот именно его нам и нужно отлавливать. Добавляем нужный хэндлер:

```
@dp.callback_query_handler(text='button1')
async def process_callback_button1(callback_query:
types.CallbackQuery):
   await bot.answer_callback_query(callback_query.id)
   await bot.send_message(callback_query.from_user.id, 'Нажата
первая кнопка!')
```




Хорошим тоном будет отвечать на все колбеки - для этого есть метод answerCallbackQuery. В документации сказано, что ответ ожидается клиентом, и дать его нужно обязательно, даже если вы не собираетесь передавать что-либо. Этого времени предостаточно, чтобы собрать

необходимые данные и прислать их. При этом во время ожидания на кнопке будут крутиться часики, показывающие, то клиент ждет ответа, поэтому если мы не хотим, чтобы пользователь наблюдал их, то нужно не забывать отвечать. Обязательный аргумент - айди запроса, на который мы отвечаем.

Создадим еще клавиатуру.

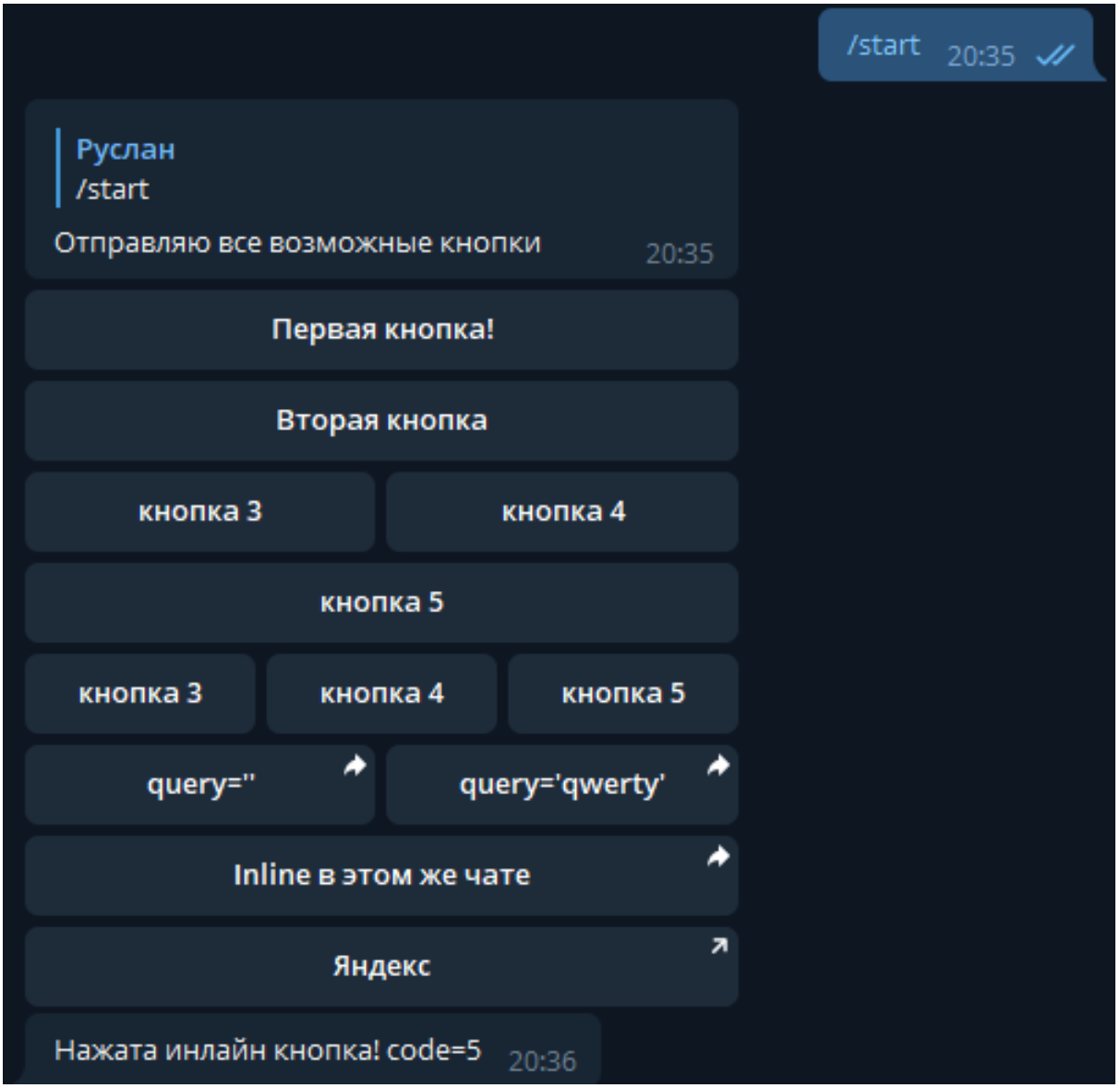
```
from aiogram.types import InlineKeyboardButton,
InlineKeyboardMarkup

inline_btn_1 = InlineKeyboardButton('Первая кнопка!',
callback_data='button1')
inline_kb_full =
InlineKeyboardMarkup(row_width=2).add(inline_btn_1)
inline_kb_full.add(InlineKeyboardButton('Вторая кнопка',
callback_data='btn2'))
inline_btn_3 = InlineKeyboardButton('кнопка 3',
callback_data='btn3')
inline_btn_4 = InlineKeyboardButton('кнопка 4',
callback_data='btn4')
inline_btn_5 = InlineKeyboardButton('кнопка 5',
callback_data='btn5')
inline_kb_full.add(inline_btn_3, inline_btn_4, inline_btn_5)
inline_kb_full.row(inline_btn_3, inline_btn_4, inline_btn_5)
inline_kb_full.insert(InlineKeyboardButton("query=''",
switch_inline_query=''))
inline_kb_full.insert(InlineKeyboardButton("query='qwerty'",
switch_inline_query='qwerty'))
inline_kb_full.insert(InlineKeyboardButton("Inline в этом же
чате", switch_inline_query_current_chat='wasd'))
inline_kb_full.add(InlineKeyboardButton('Яндекс',
url='https://www.yandex.ru'))
```

```
@dp.callback_query_handler(Text(startswith=('btn')))
async def process_callback_kb1btn1(callback_query:
types.CallbackQuery):
   code = callback_query.data[-1]
   if code.isdigit():
       code = int(code)
   if code == 2:
       await bot.answer_callback_query(callback_query.id,
text='Нажата вторая кнопка')
   elif code == 5:
       await bot.answer_callback_query(
           callback_query.id,
           text='Нажата кнопка с номером 5.\nA этот текст может
быть длиной до 200 символов', show_alert=True)
   else:
```

```
        await bot.answer_callback_query(callback_query.id)
    await bot.send_message(callback_query.from_user.id, f'Нажата
инлайн кнопка! code={code}')


@dp.message_handler(commands=['start'])
async def process_command_2(message: types.Message):
    await message.reply("Отправляю все возможные кнопки",
reply_markup=inline_kb_full)
```




Пройдёмся по строчкам по порядку, чтобы не осталось вопросов:

- мы создаём клавиатуру типа InlineKeyboardMarkup, указываем, что ширина строки должна быть не больше двух (напомню, что это не распространяется на метод row) и сразу добавляем туда уже готовую кнопку

- далее добавляем кнопку, у которой указываем другие данные в параметре callback_data
- следом генерируем три новые кнопки и добавляем их дважды. Сначала методом add, затем через row. И так как ширина клавиатуры равна двум, то в первом случае происходит перенос третьей кнопки, во втором случае нет
- затем добавляем кнопки, у которых указываем уже не callback_data, а другие параметры. То, что мы добавим в switch_inline_query, будет автоматически использовано при нажатии кнопки: пользователю предложат выбрать чат, а там вызовется инлайн режим этого бота (в поле ввода сообщения добавится юзернейм бота), следом через пробел будет прописано то, что мы указали. Параметр может принимать пустую строку, тогда инлайн режим запустится без какого-либо запроса, но если будет указан текст, то он и добавится
- при использовании switch_inline_query_current_chat произойдёт ровно то же, что и в предыдущем пункте, но без выбора чата, а запустится в текущем (было сложно догадаться по названию, я знаю)
- ну и последний параметр url - добавляем ссылку на внешний ресурс, либо диплинк в самом Телеграме

Так как параметр клавиатуры row_width равен двум, то кнопки автоматически расставились соответствующе. Рассмотрим реакцию на кнопки по порядку: При нажатии первой срабатывает наш первый колбек, так как не важно, в какую клавиатуру добавлена кнопка, важно, какая у неё callback_data . Поэтому добавлять инлайн кнопку можно сколько угодно раз в любые инлайн клавиатуры.
Кнопки со второй по пятую имеют схожую структуру в callback_data, поэтому внутри хэндлера проверяем, какой код у нажатой кнопки и:

- если 2, то отвечаем на запрос и передаем информационное сообщение. Аргумент text это текст ответа на запрос. По умолчанию он будет показан вверху чата и сам скроется через пару секунд.
- если 5, то отвечаем так же, но указываем show_alert=True, таким образом мы сообщаем клиенту, что нужно показать окошко с текстом
- в ином случае просто отвечаем на колбек

В отличие от обычных кнопок, инлайновые цепляются не к низу экрана, а к сообщению, с которым были отправлены. Давайте разберем на практике более подробно еще раз два типа кнопок: URL и Callback.(Существуют еще Login- и Pay-кнопки)

Создадим шаблон бота

```
import logging

from aiogram import Bot, Dispatcher, executor, types
from config import BOT_TOKEN
```

```
# Объект бота
bot = Bot(token=BOT_TOKEN, parse_mode=types.ParseMode.HTML)
# Диспетчер для бота
dp = Dispatcher(bot)
# Включаем логирование, чтобы не пропустить важные сообщения
logging.basicConfig(level=logging.INFO)

if __name__ == "__main__":
    # Запуск бота
    executor.start_polling(dp, skip_updates=True)
```

Самые простые инлайн-кнопки относятся к типу URL, т.е. «ссылка». Поддерживаются только протоколы HTTP(S) и tg:

```
@dp.message_handler(commands="inline_url")
async def cmd_inline_url(message: types.Message):
    buttons = [
        types.InlineKeyboardButton(text="Yandex",
url="https://yandex.ru"),
        types.InlineKeyboardButton(text="Оф. канал Telegram",
url="tg://resolve?domain=telegram")
    ]
    keyboard = types.InlineKeyboardMarkup(row_width=1)
    keyboard.add(*buttons)
    await message.answer("Кнопки-ссылки", reply_markup=keyboard)
```

В данном случае мы создали клавиатуру прямо в обработчике для простоты( не советую такое делать, лучше все же разбивать на отдельные модули)

Если хотите обе кнопки в ряд, то уберите row_width=1 (тогда будет использоваться значение по умолчанию 3).