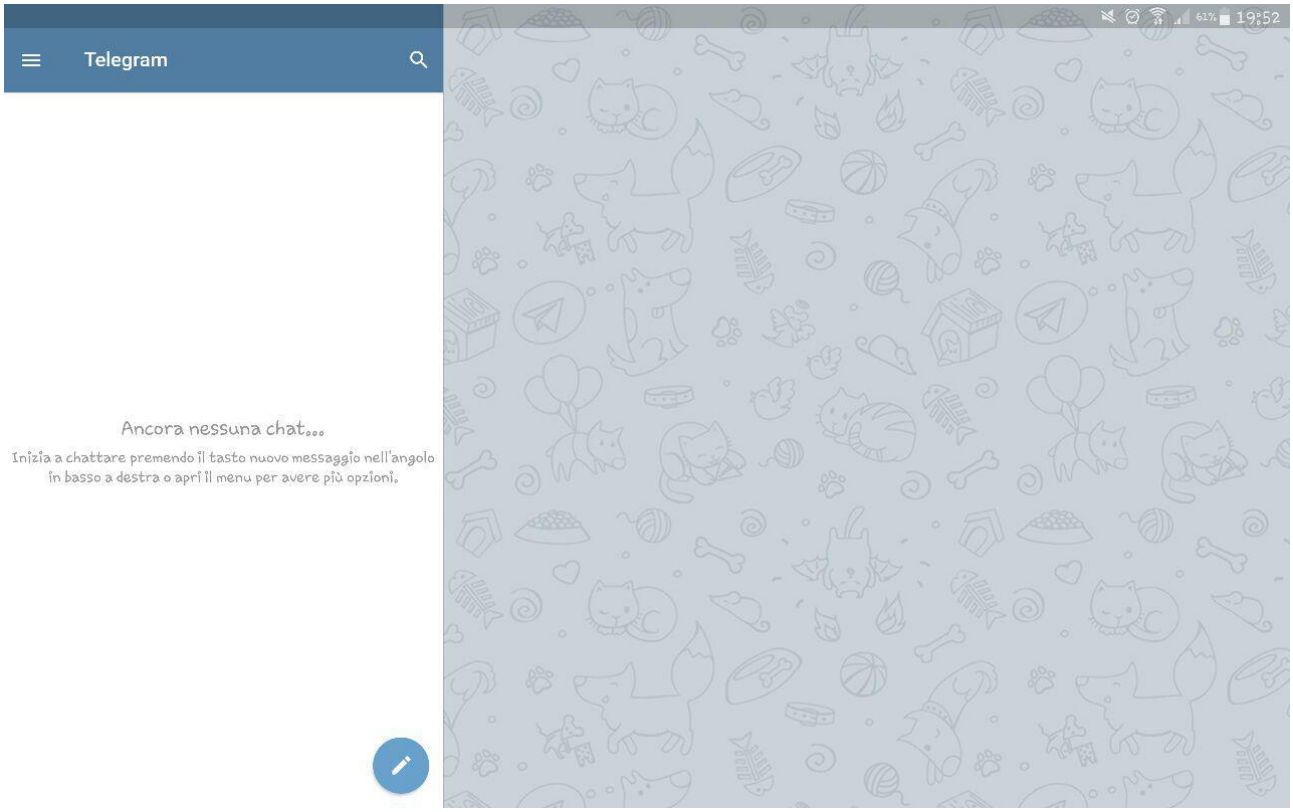
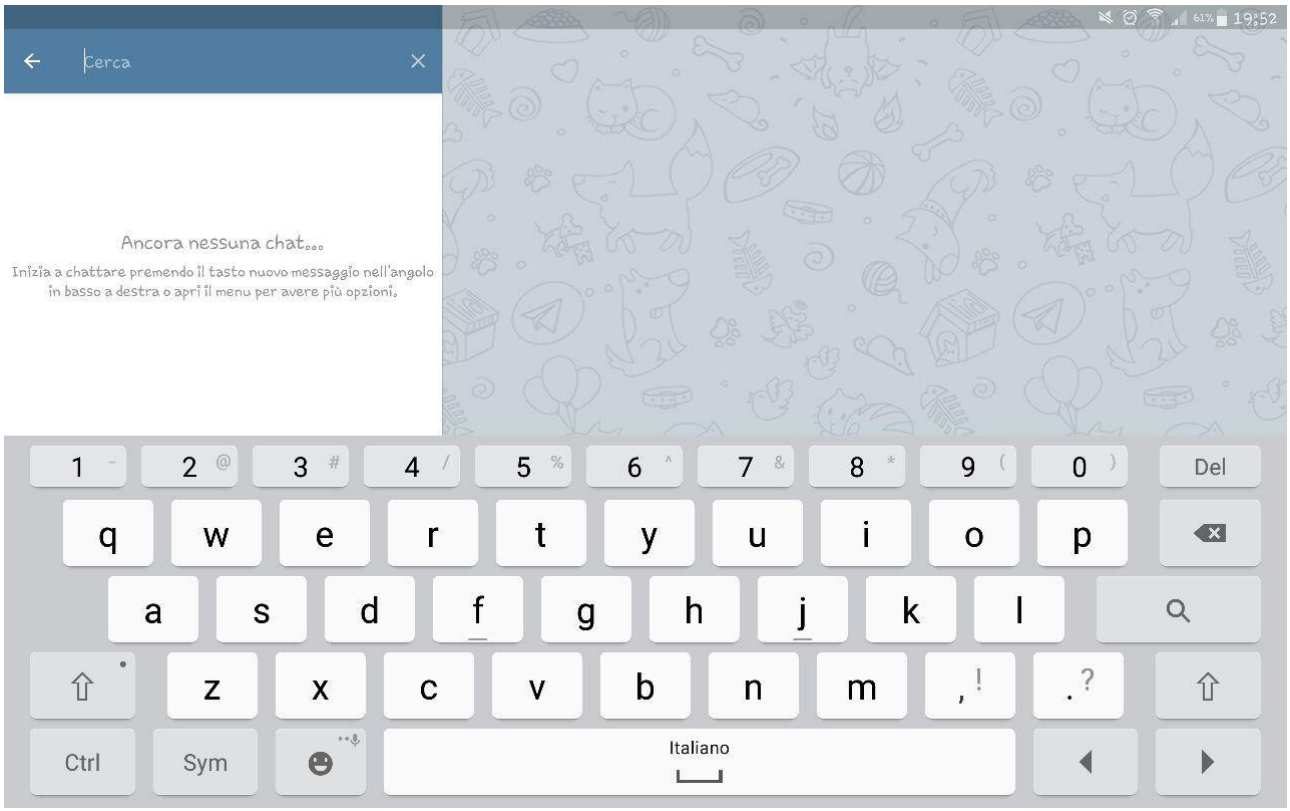
FISHGRAM

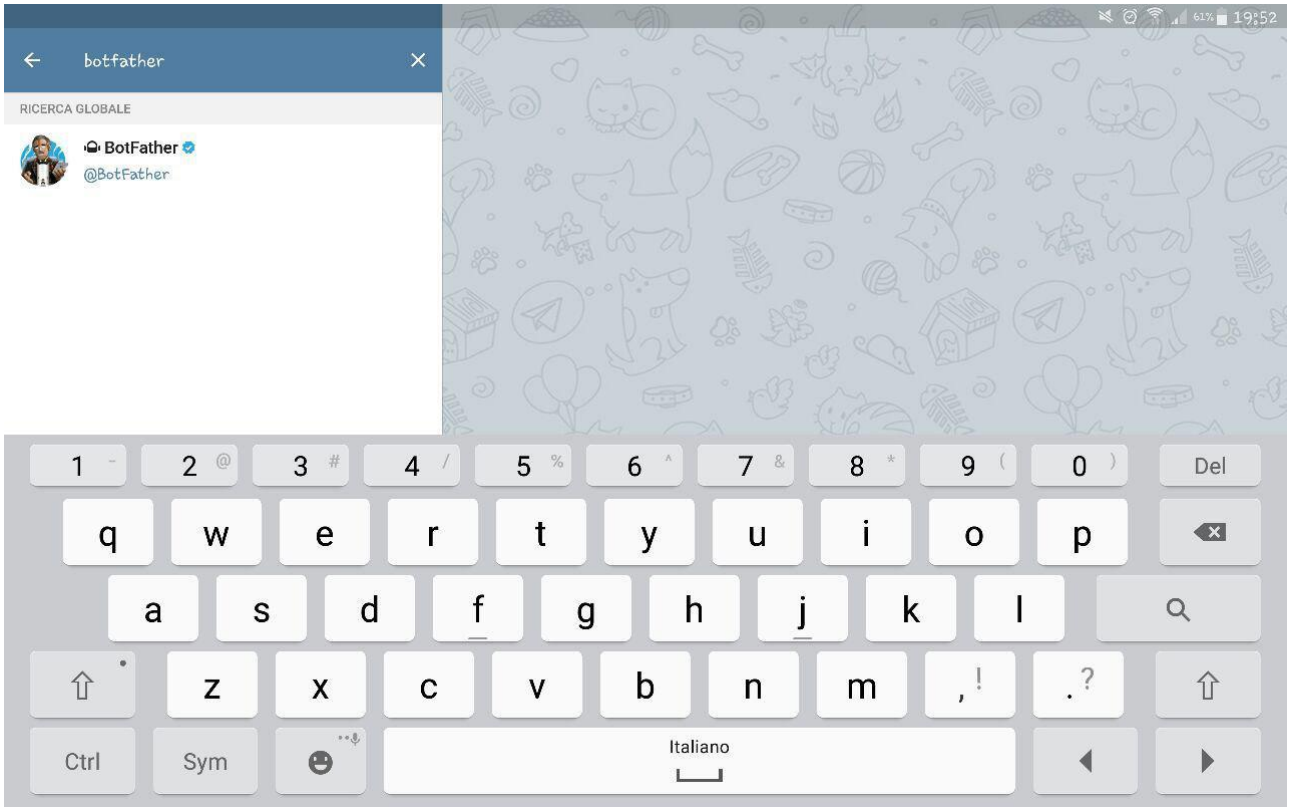
Fishgram è una libreria interessante che permette la gestione del nostro Fishino tramite un semplice messaggio su Telegram. In questo file pdf troveremo una spiegazione esauriente dell'esempio Ricevi_Invia_Messaggio.

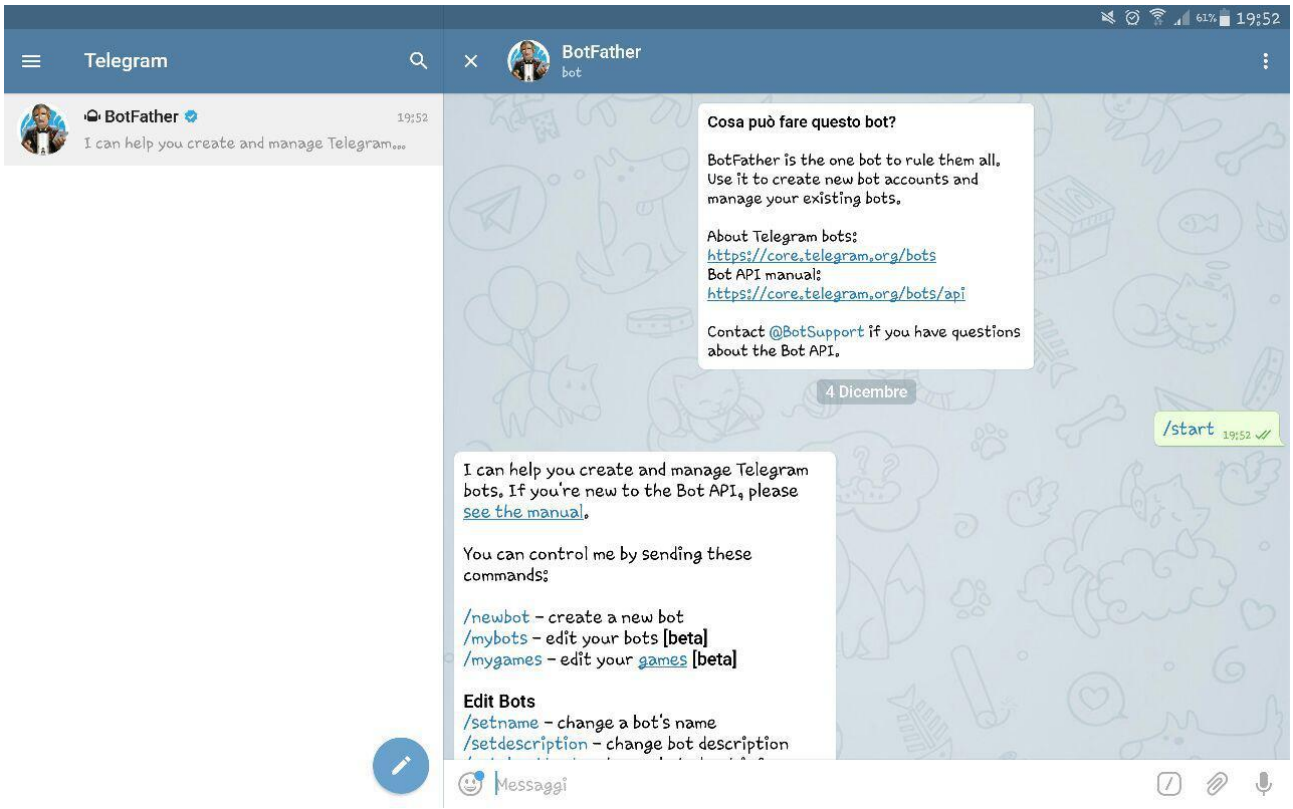
Innanzitutto dobbiamo scaricare gratuitamente Telegram sul nostro dispositivo portatile (Android – IOS – Windows – Linux – …) collegandoci sullo store oppure su https://telegram.org/.



Una volta registrati, utilizzando lo strumento di ricerca, dobbiamo cercare Botfather, avviarlo e creare il nostro bot.

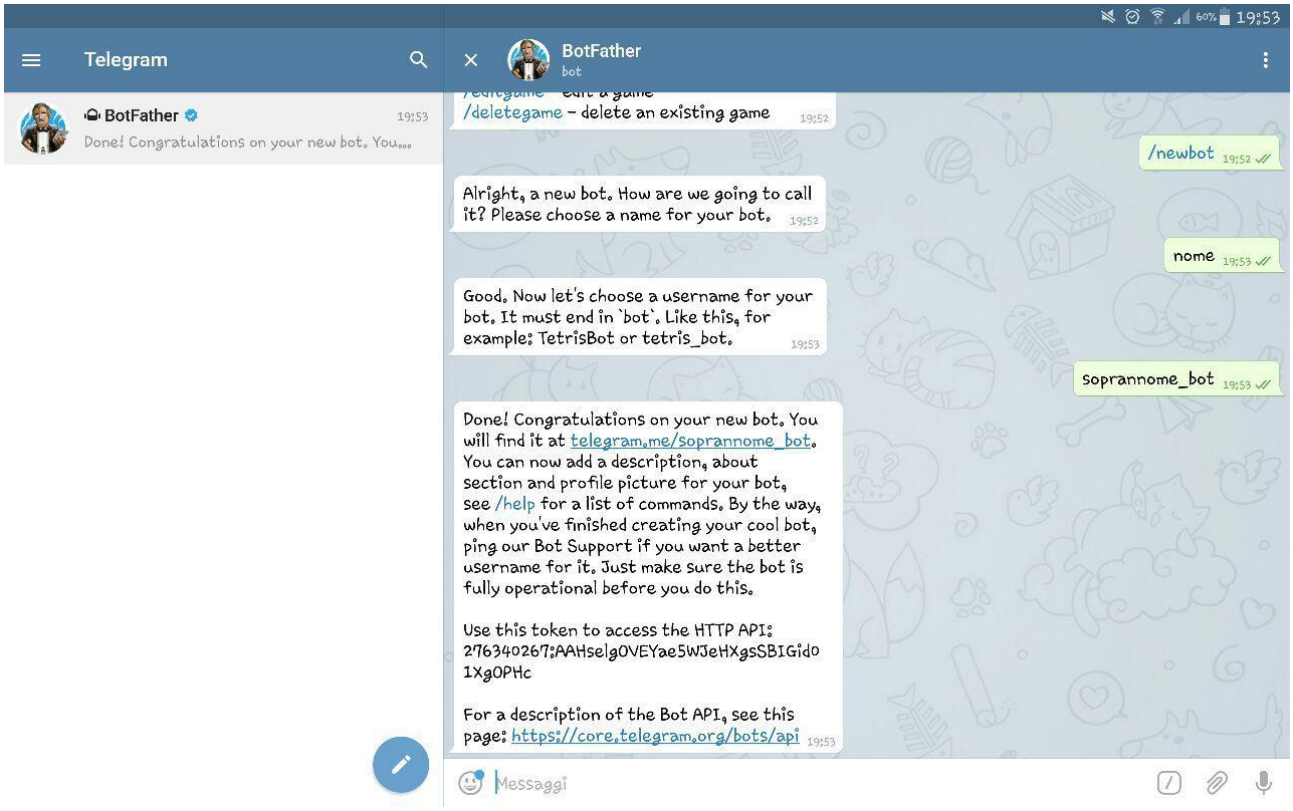

cerca
Ancora nessuna chat...
Inizia a chattare premendo il tasto nuovo messaggio nell'angolo in basso a destra o apri il menu per avere più opzioni.
Italiano


botfather
RICERCA GLOBALE
BotFather
@BotFather
Italiano



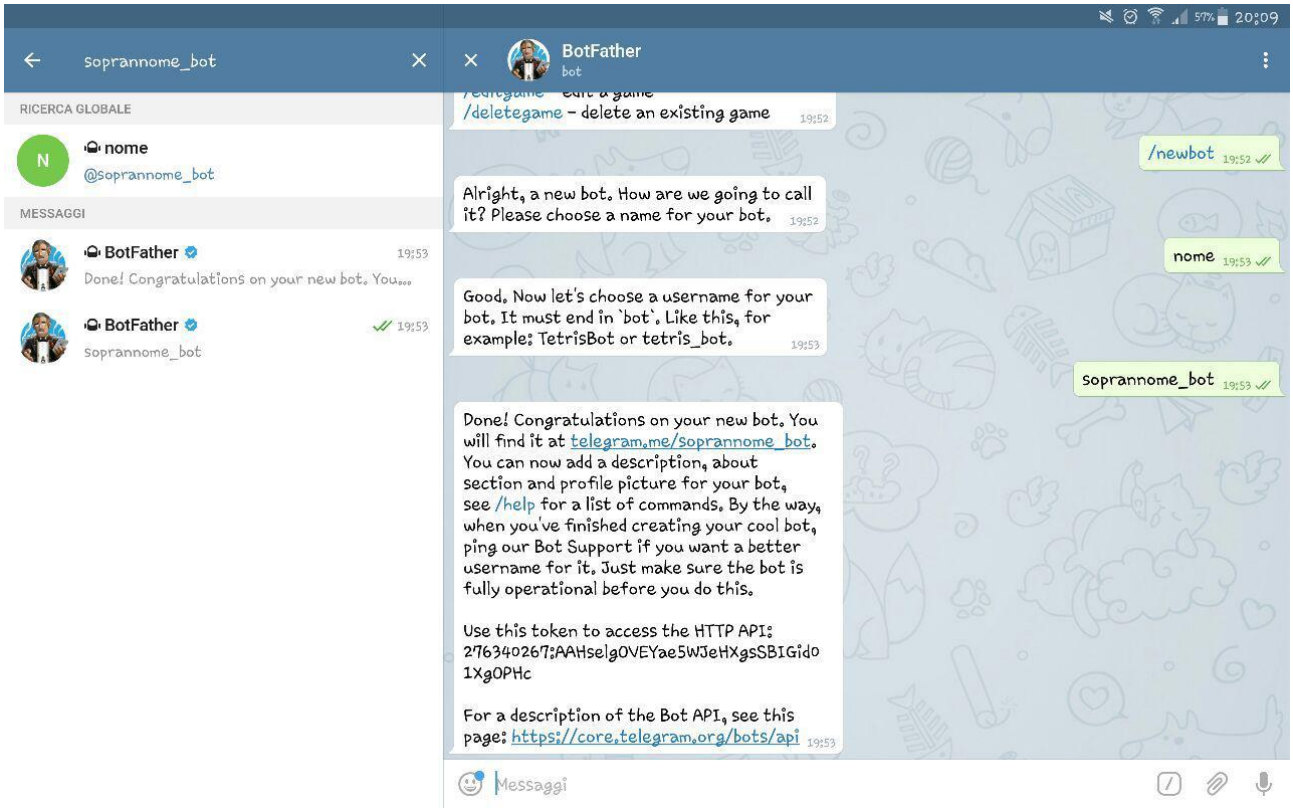
Il procedimento è molto semplice:

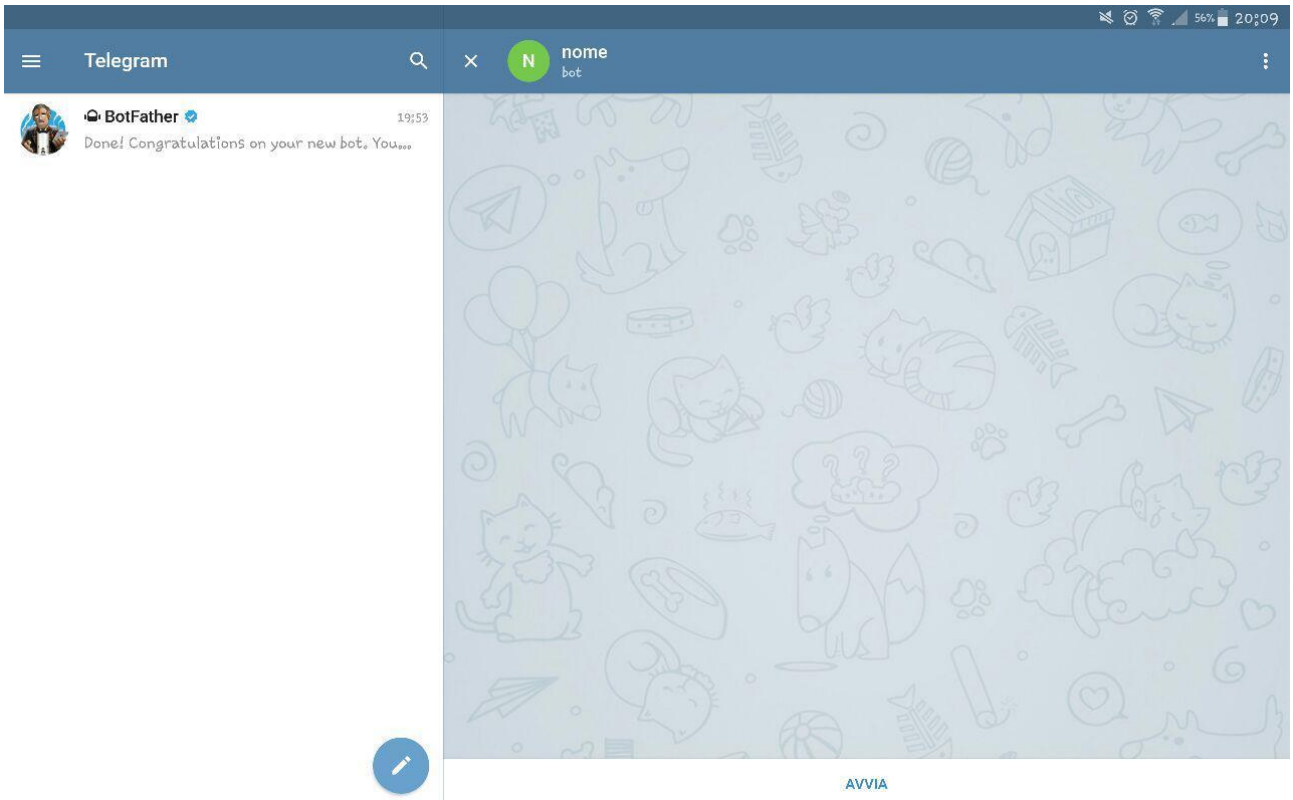
- premere su /newbot
- scegliere un nome
- scegliere un soprannome terminante in “bot”

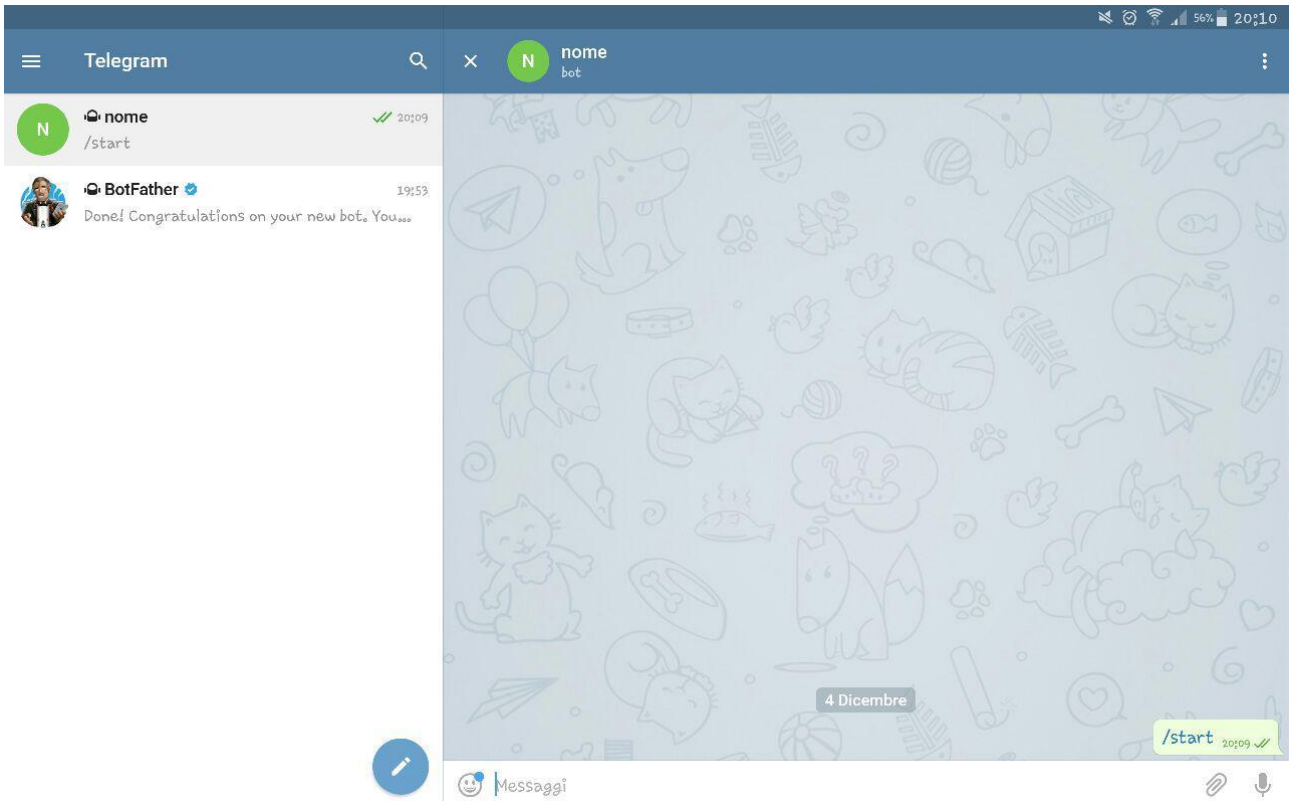


Annotiamoci l'access token, ci servirà in seguito:
276340267:AAHselgOVEYae5WJeHXgsSBIGid01XgOPHc

Adesso dobbiamo permettere al bot di comunicare con noi attivandolo. Sempre nella casella di ricerca dobbiamo trovare il nostro bot cercandolo tramite il soprannome che abbiamo scelto. Aperta la chat dobbiamo premere su avvia come abbiamo fatto per BotFather.







Così facendo saremo in grado di comunicare con il bot e lui sarà in grado di comunicare con noi.

Ai più attenti non sarà sfuggito sicuramente un importante dettaglio: se per attivare il nostro bot è stato sufficiente cercarlo e premere avvia significa che chiunque può farlo. Questo è vero. Ricordiamo però che solo chi ha creato il bot può modificarne le proprietà tramite BotFather oppure conoscerne l'access token. Inoltre ogni messaggio inviato al Bot viene contrassegnato da un id utente al quale abbiamo accesso. Perciò è possibile inserire un controllo nel codice per eseguire solo le istruzioni dell'id autorizzato.
I restanti messaggi verranno letti ma ignorati.

Addentriamoci quindi nei meandri del codice.

Dopo aver incluso le librerie necessarie e dopo aver impostato SSID, password e varie per collegarci a una rete Wifi troviamo:

```
FishinoSecureClient client;
const char accessToken[] = "*********:**********************************";
```

La prima linea di codice crea un oggetto client che supporta le connessioni sicure (https). Nella seconda invece dobbiamo inserire l'access token che poco fa abbiamo trovato. Nel nostro caso la seconda riga diventerà:

```
const char accessToken[] = "276340267:AAHselgOVEYae5WJeHXgsSBIGid01XgOPHc ";
```

Dopo e solo dopo aver inserito quest'ultime due linee di codice inseriamo:

```
Fishgram bot(accessToken, client);
```

Questa istruzione crea un'istanza della classe Fishgram e necessita di due parametri: l'access token sotto forma di const char e l'oggetto client che abbiamo creato poco fa. In questo caso l'istanza ha come nome "bot", ma naturalmente può avere il nome che meglio preferite.
Troviamo quindi l'unico id autorizzato, anche se non siamo in alcun modo limitati ad averne solo uno.

```
long id_giovanni = ********;
```

Per conoscere il nostro id è sufficiente aprire un qualsiasi browser e inserire la seguente stringa (avendo cura di inserire il vostro access token) e premere invio:
https://api.telegram.org/access_token/getUpdates

Quella per il bot che ho creato in precedenza sarà:
https://api.telegram.org/bot304107554:AAHpD4LXvsGid9mQLMpQlqEcMqLA7iDa5TE/getUpdates

Se tutto è andato liscio dovrebbe comparirvi una stringa simile a questa:

{"ok":true,"result":[{"update_id":638542571,
"message":{"message_id":2,"from":{"id":********,"first_name":"Andrea Simone","last_name":"Costa"},"chat":{"id":°°°°°°°°,"first_name":"Andrea Simone","last_name":"Costa","type":"private"},"date":1480927961,"text":"/start","entities":[{"type":"bot_command","offset":0,"length":6}]}}]}

Le 8 cifre coperte dagli asterischi * formano il vostro id.

Se per qualche motivo visualizzate una stringa come questa:
{"ok":true,"result":[]}
è probabile che non avete avviato il bot dal vostro dispositivo mobile. Avviatelo, oppure scrivete qualsiasi cosa e premete invio.
Dopodiché provate ad aggiornare la pagina del browser.

È degno di nota vedere come il bot di Telegram non ha accesso al nostro numero di telefono. Quando ci registriamo a Telegram, dopo aver inserito il nostro nome e cognome, ci viene associato un id unico che permetterà di identificarci univocamente nell'universo di Telegram.
Questo id è visibile al bot.

Adesso possiamo analizzare la void setup e la void loop del nostro codice, per poi incentrarci sulle due void esterne.

L'unica istruzione rilevante nel setup è:

```
resetSetESP();
```

Questa procedura gestisce le impostazioni del modulo ESP e permette ad esso, dopo un reset iniziale, di collegarsi alla nostra rete wifi. Più avanti vedremo perché queste istruzioni non sono presenti solo nel setup, quindi comprenderemo il motivo per cui è necessario inserirle in una void esterna.

Nel loop innanzitutto troviamo un delay di 2000 ms per evitare di inviare richieste ai server Telegram con troppa frequenza.
Dopo la chiamata a timeEsp(), la seconda procedura esterna, troviamo:

```
oldestMessage messaggioRicevuto = bot.getOldestMessage();
```

Con questa istruzione creiamo una variabile messaggioRicevuto di tipo oldestMessage. Questa non è altro che una semplice struttura utilizzata internamente dalla libreria Fishgram, nella quale troviamo tutte le informazioni necessarie del messaggio più datato, ma non ancora processato, che un utente ha scritto al nostro bot tramite Telegram.
La chiamata del metodo getOldestMessage infatti restiituisce proprio un oggetto oldestMessage, oggetto che noi memorizziamo dentro alla variabile messaggioRicevuto.

Una struttura oldestMessage ha quattro dati membro:

-isEmpty è una variabile booleana che è true se il messaggio JSON ricevuto da Telegram è una stringa vuota o non valida, altrimenti è false. In altre parole, dato che Fishino ripetutamente interpella Telegram, la maggior parte delle risposte che riceve non conterranno alcun messaggio. Quindi un controllo tramite questa variabile è utile per agire solo nel caso in cui effettivamente riceviamo qualcosa.

-sender_id è una stringa con l'id del mittente del messaggio.

-chat_id è una stringa con l'id della chat dalla quale è partito il messaggio verso il bot.

-text è una stringa contenente il testo ricevuto. Bisogna notare che qualsiasi messaggio inviato al nostro bot viene contrassegnato da una virgoletta “ iniziale e da una virgoletta ” finale. Se ad esempio vogliamo che tramite il comando pin3On il nostro Fishino attivi il pin 3, possiamo verificare l’uguaglianza tra la stringa text e una stringa di nostra scelta. Per quanto detto sopra quest’ultima stringa non sarà pin3On, bensì sarà “pin3On”.

Proseguendo nel loop troviamo:

```
unsigned long senderID = (messaggioRicevuto.sender_id).toInt()
```

Questa istruzione crea una variabile long senderID nella quale memorizziamo il risultato della conversione da stringa ad intero dell’id del mittente. Questo id, tramite getOldestUpdates, ci viene restituito sotto forma di stringa (`messaggioRicevuto.sender_id` è una stringa) e noi lo convertiamo in long per poterlo confrontare successivamente con un'altra variabile long. Quale? Quella che contiene l’id da noi autorizzato.

A questo punto troviamo un costrutto if, il quale verifica due condizioni:

- `(!messaggioRicevuto.isEmpty)`

ci assicuriamo che il messaggio JSON proveniente da Telegram contenga un messaggio non vuoto e valido.

- `(senderID == id_giovanni)`

infine ci assicuriamo che l’id del mittente sia autorizzato.

Se queste condizioni vengono soddisfatte stampiamo sulla seriale il testo del messaggio inviato dall’id autorizzato tramite:

```
Serial.println(messaggioRicevuto.text);
```

Dopo aver creato una stringa da inviare tramite:

```
const char rispostaBot[] = "Ciao, come stai?";
```

Rispondiamo al messaggio ricevuto grazie al metodo sendMessage:

```
bot.sendMessage(rispostaBot, messaggioRicevuto.sender_id);

bot.sendMessage(messaggioRicevuto.text, messaggioRicevuto.sender_id);
```

I parametri richiesti da questo metodo sono il testo da inviare sotto forma di const char o di stringa e l'id del destinatario. In questo esempio rispondiamo all'id che ha inviato il messaggio prima con un testo di nostra creazione e poi con lo stesso testo ricevuto poco prima.

Dato che il codice presente nella libreria non è ancora perfettamente stabile può succedere che Fishino vada in blocco. La causa di questo blocco risiede comunque in un difetto di comunicazione con Telegram, perciò è necessario riavviare solo il modulo ESP. Per questo le istruzioni per resettarlo e impostarlo sono inserite nella procedura `resetSetESP();`, in modo da essere richiamate quando è necessario.
È risultato conveniente impostare un reset a intervalli periodici tramite la funzione millis(). Questo reset è gestito dalla void `timeEsp()`, la quale consiglio di non eliminare ne modificare.
L'unico cambiamento utile potrebbe essere quello dell'intervallo di tempo tra due reset. Di default è impostato a 600'000 millisecondi, ovvero dieci minuti, tramite la variabile `intervallo`.