**Building a chat bot with Home Assistant, AppDaemon and Telegram**

**Setting up a new Telegram bot**

To start, we need to make a Telegram bot to link with HA.

1. Start a new conversation with the [BotFather](https://telegram.me/botfather). This is a special bot that can be used to manage and create bots.
2. Use the command /newbot to start the process of making a bot.
3. Fill in the name of your bot (this is the display name).
4. Fill in the username of your bot (this has to be in the format of <your\_bot\_name>bot).
5. When all is done you will receive a HTTP API token from the BotFather. This is required to connect the bot to HA.
6. After creating your bot you can start chatting with it. Note that a user must always start a chat with a bot; a bot cannot start a conversation on its own.

**Connecting a Telegram bot to Home Assistant**

To start connecting our new Telegram bot to HA we need to enter our credentials for accessing the bot. We do this by adding a few lines to secrets.yaml. If we do not want to use the secrets file we can also skip this step and enter the credentials directly.

telegram\_bot\_key: <your api key from the BotFather>

telegram\_bot\_main\_chat\_id: <your chat id>

telegram\_bot\_chat\_ids:

- <your chat id>

- <optional: other chat ids you want your bot to communicate with>

The chat id is the unique number that identifies our Telegram account. If we do not know our chat id we can send /getid to one of the ‘id’-bots.

The next step is to load the Telegram bot in HA by adding the following lines to our configuration.yaml

telegram\_bot:

- platform: polling

api\_key: **!secret** telegram\_bot\_key

allowed\_chat\_ids: **!secret** telegram\_bot\_chat\_ids

I am using the polling method for the Telegram bot. Using this method we do not have to open up our Home Assistant instance to the outside world.

We can also link the bot to the [Notify component](https://www.home-assistant.io/components/notify/) of home assistant:

notify:

- name: telegram

platform: telegram

chat\_id: **!secret** telegram\_bot\_main\_chat\_id

**Basis of an AppDaemon Telegram app**

[AppDaemon](https://www.home-assistant.io/docs/ecosystem/appdaemon/) is a great tool for building our Telegram bot as we can leverage the flexibility of Python. Alternatively, it is also possible to build your bot totally using yaml files, but that will take a bit more effort.

Start by adding a new instance to our apps.yaml configuration for AppDaemon:

bot:

module: telegram

class: TelegramBot

Then create the associated python file (telegram.py). To get started, copy the following to the file to use as wireframe:

import appdaemon.plugins.hass.hassapi **as** hass

**class** **TelegramBot**(hass.Hass):

**def** **initialize**(self):

*# Start listening for Telegram updates*

self.listen\_event(self.receive\_telegram\_text, 'telegram\_text')

**def** **receive\_telegram\_text**(self, event\_id, payload\_event, **\***args):

*# Do something with the text*

user\_id **=** payload\_event['user\_id']

message **=** payload\_event['text']

Save the file and the apps should automatically start. The app starts listening to telegram\_text events; these are triggered when the bot receives a message.

**Saying Hi!**

The first feature we can build in to the bot is to say a welcome message. However, it would be kind of annoying if the bot said “Hi” every time we send a new message to the bot. To solve this I only send a greeting message at most once an hour.

Extend the initialize function of the class to keep track of your conversations using a dictionary. We will store the last time a user sent a message in this dictionary.

**def** **initialize**(self):

self.time\_between\_conversations **=** 60 **\*** 60

self.last\_conversation **=** {}

A new function is added to the class that sends a greeting message. It checks whether the last message was sent longer than 60 minutes and, if so, sends a greeting message.

**def** **greet\_user\_if\_new\_conversation**(self, user\_id, user\_name):

"""Say hi when there is a new conversation"""

*# For new users automatically set the time to zero.*

**if** user\_id **not** **in** self.last\_conversation:

self.last\_conversation[user\_id] **=** 0

*# Compute the time difference in seconds since the last message.*

time\_diff **=** time.time() **-** self.last\_conversation[user\_id]

**if** time\_diff **>** self.time\_between\_conversations:

msg **=** f"Hi {user\_name}"

*# Send a message to the user.*

self.call\_service('telegram\_bot/send\_message',

target**=**user\_id,

message**=**msg)

self.last\_conversation[user\_id] **=** time.time()

This new function is then called each time we receive a text:

**def** **receive\_telegram\_text**(self, event\_id, payload\_event, **\***args):

self.log(payload\_event)

user\_id **=** payload\_event['user\_id']

message **=** payload\_event['text']

self.greet\_user\_if\_new\_conversation(user\_id, payload\_event['from\_first'])

The from\_first payload variable should contain our first name. Saying something to our bot should now result in a simple greeting.

**Asking for the temperature**

Now let’s do something usefull. I would like to keep track of the current temperature in my house. To facilitate this I created a small function that reports this. However, I only want my bot to tell me the temperature when I ask for it. So we need to do some very simple text parsing. I would like the bot to react to simple questions like:

*Is it cold outside?*

*What is the temperature?*

*Is it warm at home?*

This can be done with some simple rules in python.

**if** any((

'temperature' **in** message,

('warm' **in** message **or** 'cold' **in** message) **and** ('house' **in** message **or** 'inside' **in** message **or** 'outside' **in** message),

('warm' **in** message **or** 'cold' **in** message) **and** ('house' **in** message **or** 'inside' **in** message **or** 'outside' **in** message),

)):

*# Report the temperature*

self.send\_house\_temperature(user\_id)

**def** **send\_house\_temperature**(self, user\_id):

"""Send information about the temperature in the house"""

sensors **=** {

'The outside temperature is {} °C.': 'sensor.temp\_1', *# Replace this with your sensor's entity id*

'It is {} °C in the livingroom.': 'sensor.temp\_2', *# Replace this with your sensor's entity id*

'The temperature of the livingroom is {} °C.': 'sensor.temp\_2' *# Replace this with your sensor's entity id*

}

start\_messages **=** [

'Let me see.',

'These are the temperatures currently.',

'I have read the sensors for you.'

]

msg **=** random.choice(start\_messages)

*# Read all sensors.*

**for** template, sensor **in** sensors.items():

temp **=** self.get\_state(sensor)

msg **+=** ' ' **+** template.format(temp)

self.call\_service('telegram\_bot/send\_message',

target**=**user\_id,

message**=**msg)

In this new function I read out different sensors around my home. In this case measuring the outside temperature, and the temperature inside my livingroom and bedroom. A message is then constructed using a randomly chosen start sentence and the sensor readings.

This is the quick demonstration of making a Telegram bot, connecting it to Home Assistant and using AppDaemon to send messages.