Luigi De Russis, Alberto Monge Roffarello

LAB 3 – PYTHON INTERMEDIATE

GETTING STARTED

Recap:

1. *Fork your own copy of the Git repository associated with this lab (https://github.com/AmI-2019/python-lab3) to your personal GitHub space*
2. *Open PyCharm Professional and select Checkout from Version Control > Git in the "Welcome to PyCharm" window, to clone your (forked) repository*
3. *Fill the requested fields (repository URL, location on disk, …) and press the "Clone" button*
4. *Once the project is open, you can create a new Python file by right clicking on the project name (Project tab, on the left) and selecting New > Python File*
5. *Commit and push the changes you made back to GitHub, from the VCS menu in PyCharm*

The goal of this set of exercises is to convert the third exercise of the previous laboratory[1] in a Telegram bot.

Before starting the exercises, you should create a new bot account:

1. If you already have a Telegram account go to point number 2, otherwise:
    a. Install Telegram (web version at http://web.telegram.org)
    b. Register your phone number (insert an existing number because the system will send you an activation number)
2. Open Telegram
3. Search for "@BotFather" and start a chat with it, by pressing the "Start" button
4. Send the message "/newbot"
5. Type the name "AmITaskList<id>Bot" (substitute <id> with your student id)
6. Type the username "AmITaskList<id>_bot" (substitute <id> with your student id)
7. BotFather will give you a TOKEN that you must use for accessing the Telegram API

In the following, we will use <TOKEN> to indicate the token provided at step 7.

Use the library "python-telegram-bot" to create a Telegram bot in Python. Documentation and tutorials are reachable from https://github.com/python-telegram-bot/python-telegram-bot.

[1] A possible solution to the exercise can be found at https://github.com/AmI-2019/python-lab2/tree/solution (*todo_manager_ex3.py*).

Luigi De Russis, Alberto Monge Roffarello

EXERCISE – TELEGRAM BOT: TODO LIST MANAGER

Modify the program developed in the third exercise of the previous laboratory so that a Telegram bot is run at startup. **Before running** the bot, the program should **load the task list** contained in the file *task_list.txt* (you can get the file from the GitHub repository of this lab). Then, **every time the list is changed**, the program should **save the new list** to the file.

The bot should **accept only commands**, so, if the user sends any other message, the bot will answer with an error (e.g., "I'm sorry, I can't do that.").

Finally, the bot should provide the following **4 commands**:

1. /showTasks
2. /newTask <task to add>
3. /removeTask <task to remove>
4. /removeAllTasks <substring to use to remove all the tasks that contain it>

The following subsections exemplify the messages that should be exchanged between the user and the bot for each command.

/showTasks - show all the existing tasks in alphabetic order

User: /showTasks
Bot: Nothing to do, here! **OR** ["book summer holidays", "buy a new mouse", "call Giovanni for AmI project organization", "find a present for Angelina's birthday", "organize mega party (last week of April)", "whatsapp Mary for a coffee"]

/newTask - insert a new task

User: /newtask *wash the car*
Bot: The new task was successfully added to the list!
Suggestion: to handle command parameters (e.g., "wash the car") you must specify the option "*pass_args=True*" in the CommandHandler declaration.

For more details:
http://python-telegram-bot.readthedocs.io/en/latest/telegram.ext.commandhandler.html?highlight=CommandHandler

/removeTask: remove a task (by typing its content, exactly)

User: /removeTask *wash the car*
Bot: The task was successfully deleted! **OR** The task you specified is not in the list!

/removeAllTasks: remove all the existing tasks that contain a provided string

User: /removeAllTasks *wash*
Bot: The elements "wash the car" and “wash the bike” were successfully removed! **OR** I did not find any task to delete!