Habits Tracker

An Overview of Features and Functionality

By: Mohsen Nasiri - 32206338 - On Campus

International University of IU - Bad Honnef

2023.11.30

Habits Tracker: Introduction

Welcome To Habits Tracker!

We are here to assist you to improve your habits.

What does the Habit Tracker do?

The Habit Tracker is a user-friendly program that helps you in tracking and analyzing your personal habits. It also helps you stay motivated and on tack with the help of its streak system.

Habits Tracker: Key Features

Simplicity:

- With easy to use commands to interact with Habits Tracker.
- With the user-friendly interface, track habits effortlessly.

Effectiveness and Efficiency:

- Monitor various aspects, from habit completion to streaks and your all time best records.
- Receive instant feedback in real-time on your progress for continuous motivation.

Habits Tracker: Why to Use?

Why Use the Habit Tracker?

- For your personal development by cultivating positive habits and achieving your goals.

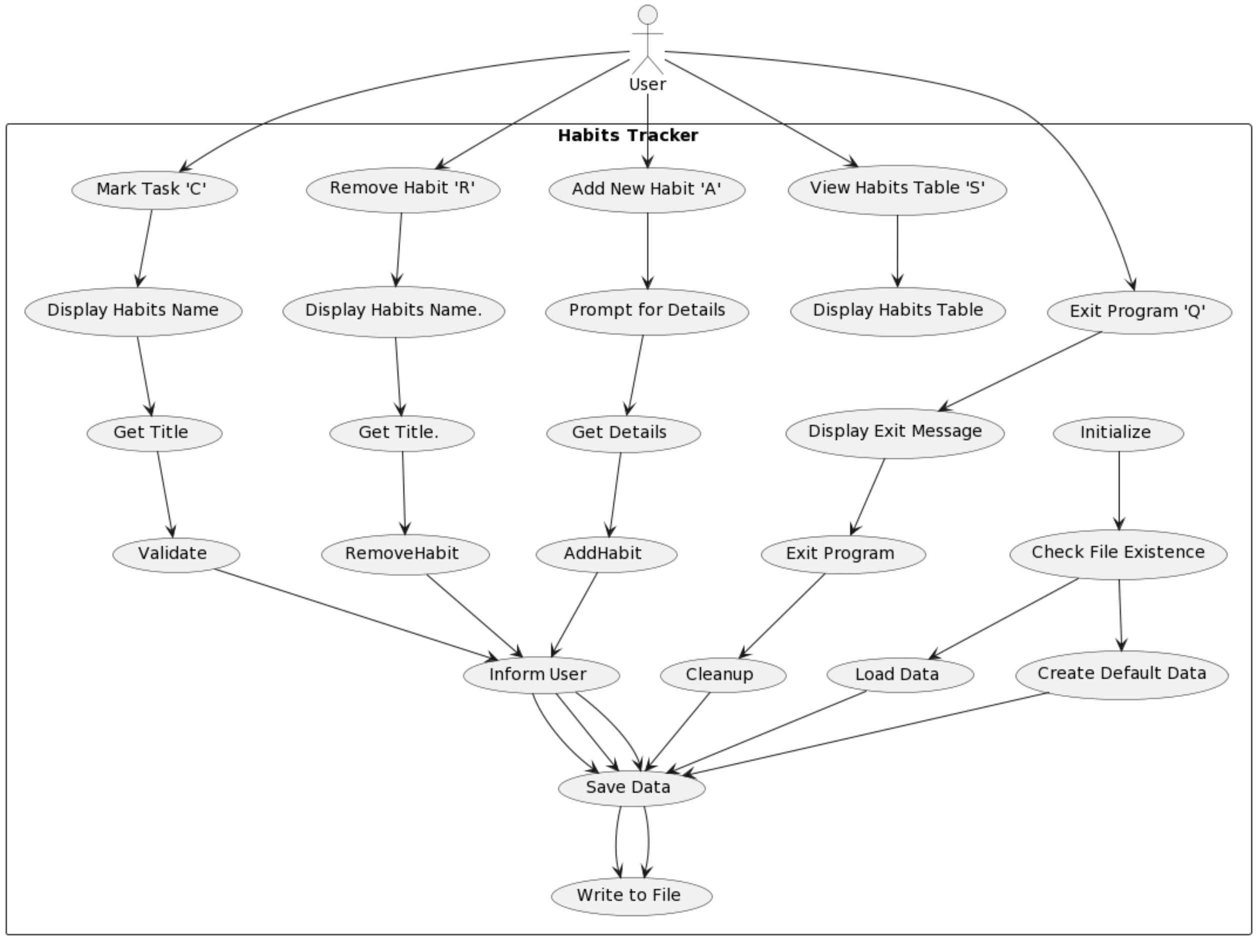
- Because its easy! Monitor your daily habits through a our interface and stay motivated for your set goals.

Habits Tracker: How it works?

This UML diagram shows the final functionality of the program.

The program starts to load or create and then modify the saved or default data.

After the initialization, the user interface comes in to play with 5 available commands.



Habits Tracker: Implementation

The program was created using Python and with the help of libraries and methods it provides.

Libraries Used:

- **JSON:** Handling of data storage and retrieval in JSON format.
- **OS:** Checking file existence and managing file-related operations.
- **DateTime:** Retrieve date and time for real-time habit tracking.
- **PrettyTable:** Generating organized tables for displaying habits and their details.

Habits Tracker: Instruction Manual / Tools

For the libraries used in the program we are not required to install from a foreign source, but in case that we are needed to for the PrettyTable library:

Step 1: Open Terminal or Command Prompt.

Step 2: Type in the console: “python -m pip install --upgrade pip ”.

Step 3: Type in the console: “pip install prettyta

Step 4: Done!

Habits Tracker: Getting Started

Once you have acquired the required files to run the program, all that is left is as simple as clicking on the file named “ Main.py ”.

Or you could also:

Step 1: Open Terminal or Command Prompt.

Step 2: Run the Main Script: “ python main.py ”.

Step 3: Follow On-Screen Prompts.

Habits Tracker

Thank you for fallowing this presentation! Your participation is invaluable to us, and we deeply appreciate the time you invest in using our program.

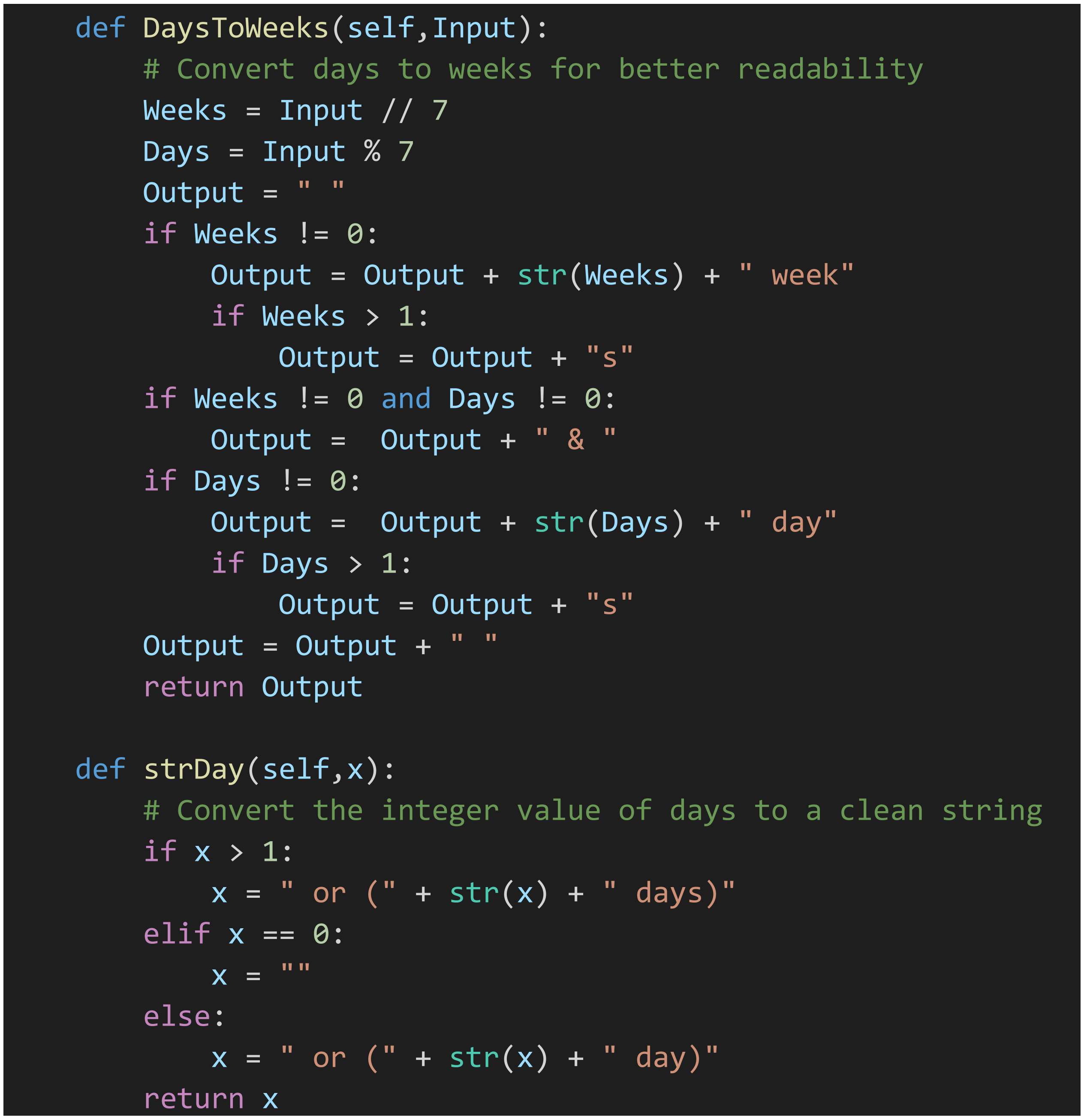
Once again, thank you for being an integral part of our community and for making Habits Tracker a meaningful tool for positive habit-building.

HabitsTracker.py Code(Backend):

This is the code that uses the libraries and tools mentioned before. It will then be used and imported into the Main.py Code the same way as a library would be used.

```python
import json
import os
import datetime
from prettytable import PrettyTable

class Tracker():
    def __init__(self):
        # Get the current date and time
        self.TodaysDate = datetime.datetime.now()

        # Check if the data file exists and laod it
        if os.path.exists('./Data.json'):
            with open('./Data.json', 'r') as file:
                self.Data = json.load(file)

            # Check habits and update their status based on
time frame
            for habit in self.Data["Habits"]:
                if self.TodaysDate >=
(datetime.datetime.strptime(self.Data["Habits"][habit]["Time"],
"%Y-%m-%d %H:%M:%S.%f") +
datetime.timedelta(days=self.Data["Habits"][habit]["Time_Frame"
])):
                    if self.Data["Habits"][habit]["Status"] ==
0 and self.Data["Habits"][habit]["Streak"] != 0:
                        print(f'\033[1m\033[91m You have lost
your {habit} { self.Data["Habits"][habit]["Streak"] } Day
Streak! \033[0m')
                        self.Data["Habits"][habit]["Streak"] =
0
```

```
                    # Update habit status and time
                    self.Data["Habits"][habit]["Status"] = 0
                    self.Data["Habits"][habit]["Time"] =
str(self.TodaysDate)
                    self.Save_Data(self.Data)

        # If the data file doesn't exist, create a default
        else:
            self.Data = {"Habits": {
                "Sleep": {"Goal": "Sleep on time",
"Time_Frame": 1, "Streak": 0, "Status": 0,"BestStreak":
0,"Time": str(self.TodaysDate)},
                "Workout": {"Goal": "Workout 1hour",
"Time_Frame": 2, "Streak": 0, "Status": 0,"BestStreak":
0,"Time": str(self.TodaysDate)},
                "Cigarettes": {"Goal": "No Smoking",
"Time_Frame": 21, "Streak": 0, "Status": 0,"BestStreak":
0,"Time": str(self.TodaysDate)},
                "Alcohol": {"Goal": "No Alcohal", "Time_Frame":
31, "Streak": 0, "Status": 0,"BestStreak": 0,"Time":
str(self.TodaysDate)},
                "Journal": {"Goal": "Journal your week",
"Time_Frame": 7, "Streak": 0, "Status": 0,"BestStreak":
0,"Time": str(self.TodaysDate)}
                }
            }
            self.Save_Data(self.Data)

    def Save_Data(self,Data):
        # Save data to the JSON file
        with open('./Data.json', 'w') as file:
            json.dump(Data, file, indent=2)

    def
AddHabit(self,Title,Goal,TimeFrame,Streak,Status,BestStreak):
        # Add a new habit to the data
        self.Data["Habits"][Title] = {
            "Goal": Goal,
            "Time_Frame": TimeFrame,
            "Streak": Streak,
            "Status": Status,
            "BestStreak": BestStreak,
            "Time": str(self.TodaysDate)
        }
        self.Save_Data(self.Data)
        print("\033[1m\033[92m Habit Added! \033[0m")

    def RemoveHabit(self,Title):
        # Remove a habit from the data
        if Title in self.Data["Habits"]:
            del self.Data["Habits"][Title]
            self.Save_Data(self.Data)
            print("\033[1m\033[92m Habit Removed! \033[0m")
        else:
            print("\033[1m\033[91m Habit Not Found! \033[0m")

    def TableofHabits(self):
        # Generate a formatted table with habit information
        self.Table = PrettyTable()
        self.Table.clear_rows()
        self.Table.field_names = ["\033[1m\033[96m Habit's
Title \033[0m", "\033[1m\033[94m Your Aim and Goal \033[0m",
"\033[1m\033[95m Time Frame \033[0m", "\033[1m\033[95m Time
Frame in days \033[0m", "\033[1m\033[91m Status \033[0m",
"\033[1m\033[92m Remaining Time \033[0m", "\033[1m\033[93m
Current Streak(In days) \033[0m","\033[1m\033[93m Best
Streak(In days) \033[0m"]
        for habit, details in
self.Data["Habits"].items():
            if details['Status'] == 0:
                Col5 = "\033[91m Incomplete \033[0m"
            else:
                Col5 = "\033[92m Complete \033[0m"
            if details['Streak'] == 0 and details['Status'] ==
0:
                Col6 = f'\033[90m
{str((datetime.datetime.strptime(details["Time"], "%Y-%m-%d
%H:%M:%S.%f") + datetime.timedelta(days=details["Time_Frame"]))
- self.TodaysDate).split(".")[0]} \033[0m'
            elif details['Streak'] != 0 and details['Status']
== 0:
                Col6 = f'\033[1m
{str((datetime.datetime.strptime(details["Time"], "%Y-%m-%d
%H:%M:%S.%f") + datetime.timedelta(days=details["Time_Frame"]))
- self.TodaysDate).split(".")[0]} \033[0m'
            else:
                Col6 = f'\033[92m
{str((datetime.datetime.strptime(details["Time"], "%Y-%m-%d
%H:%M:%S.%f") + datetime.timedelta(days=details["Time_Frame"]))
- self.TodaysDate).split(".")[0]} \033[0m'

            if details['Time_Frame'] > 1:
                Col4 = f"\033[95m {details['Time_Frame']} days
\033[0m"
            else:
                Col4 = f"\033[95m {details['Time_Frame']} day
\033[0m"
            self.Table.add_row([f"\033[96m {habit} \033[0m",
f"\033[94m {details['Goal']} \033[0m", f"\033[95m
{self.DaysToWeeks(details['Time_Frame'])} \033[0m", Col4,
Col5,Col6, f"\033[93m
{details['Streak']}{self.strDay(details['Streak'] *
details['Time_Frame'])} \033[0m",f"\033[93m
{details['BestStreak']}{self.strDay(details['BestStreak'] *
details['Time_Frame'])} \033[0m"])
        return self.Table

    def ListofHabits(self):
        # Generate a list of habit names
        self.Table = PrettyTable()
        self.Table.clear_rows()
        NamesList = []
        for habit in self.Data["Habits"]:
            NamesList.append("\033[93m " + habit + " \033[0m")
        self.Table.field_names = NamesList
        self.Table.horizontal_char = " "
        self.Table.vertical_char = " "
        self.Table.junction_char = " "
        return self.Table

    def CheckHabit(self,Title):
        # Update habit status and streaks
        if Title in self.Data["Habits"]:
            # Update habit status and streak
            if self.Data["Habits"][Title]["Status"] != 1:
                self.Data["Habits"][Title]["Status"] = 1
                self.Data["Habits"][Title]["Streak"] =
self.Data["Habits"][Title]["Streak"] + 1
                self.Data["Habits"][Title]["Time"] =
str(self.TodaysDate)
                print("\033[1m\033[92m Status Updated!
\033[0m")
            else:
                print("\033[1m\033[92m Status already Updated!
\033[0m")
            # Update the best streak if applicable
            if self.Data["Habits"][Title]["Streak"] >
self.Data["Habits"][Title]["BestStreak"]:
                self.Data["Habits"][Title]["BestStreak"] =
self.Data["Habits"][Title]["Streak"]
            self.Save_Data(self.Data)
        else:
            print("\033[1m\033[91m Habit not found! \033[0m")
```



```python
    def DaysToWeeks(self,Input):
        # Convert days to weeks for better readability
        Weeks = Input // 7
        Days = Input % 7
        Output = " "
        if Weeks != 0:
            Output = Output + str(Weeks) + " week"
            if Weeks > 1:
                Output = Output + "s"
        if Weeks != 0 and Days != 0:
            Output =  Output + " & "
        if Days != 0:
            Output =  Output + str(Days) + " day"
            if Days > 1:
                Output = Output + "s"
        Output = Output + " "
        return Output

    def strDay(self,x):
        # Convert the integer value of days to a clean string
        if x > 1:
            x = " or (" + str(x) + " days)"
        elif x == 0:
            x = ""
        else:
            x = " or (" + str(x) + " day)"
        return x
```

Main.py Code(Frontend):

This is the code or script we will use to “Run” or “Start” the Program.

It will import the code or file we created in the previous section!

```python
from HabitsTracker import Tracker

# Create an instance of the Tracker class
habits = Tracker()

# Define color codes for console output
MainColor = "\033[92m"
ResetColor = "\033[0m"
YellowColor = "\033[93m"

# Welcome message
print(f"{YellowColor} Progress is Progress no matter how small!
{ResetColor}")
print(f"{MainColor} Hello and Welcome! {YellowColor}(^_^) ")
print(f"{MainColor} You may use the commands bellow to
navigate! {ResetColor}")

# User interaction loop
while True:
    # Display the habits table
    print(habits.TableofHabits())
    print()
    # Display command options
    print(f"{MainColor} To show everything so far use:
\033[1m{YellowColor}'S' {MainColor}or
\033[1m{YellowColor}'s'{MainColor}! {ResetColor}")
```

```
    print(f"{MainColor} To mark your daily task use:
\033[1m{YellowColor}'C' {MainColor}or
\033[1m{YellowColor}'c'{MainColor}! {ResetColor}")
    print(f"{MainColor} To add a new Habit use:
\033[1m{YellowColor}'A' {MainColor}or
\033[1m{YellowColor}'a'{MainColor}! {ResetColor}")
    print(f"{MainColor} To remove a Habit use:
\033[1m{YellowColor}'R' {MainColor}or
\033[1m{YellowColor}'r'{MainColor}! {ResetColor}")
    print(f"{MainColor} To exit program use:
\033[1m{YellowColor}'Q' {MainColor}or
\033[1m{YellowColor}'q'{MainColor}! {ResetColor}")

    # Get user input
    user = input(f"{MainColor} You: {ResetColor}").lower()

    # Add a new habit
    if user == "a":
        print()
        title = input(f"{MainColor} Choose a 'Title' for this
habit: {ResetColor}")
        print()
        goal = input(f"{MainColor} Define your 'Goals and
Plans' for this habit: {ResetColor}")
        print()
        print(f"{MainColor} Define a 'Time Frame' or time cycle
for your self to complete this task")
        print(f"{MainColor} If not completed during this period
you will break the habit and your 'Streak'")
        aflag = True
        while aflag:
            timefr = input(f"{MainColor} Please define in
number of days: {ResetColor}")
            if timefr.isdigit():
                aflag = False
            else:
                print(f"{MainColor} Please enter a number!
{ResetColor}")
        print()
        print(f"{MainColor} If you have already started...")
        aflag = True
        while aflag:
            streak = input(f"{MainColor} What is your current
'Streak' for succesfully completed cycles?  {ResetColor}")
            if streak.isdigit():
                aflag = False
            else:
                print(f"{MainColor} Please enter a number!
{ResetColor}")
        print()
        stat = input(f"{MainColor} Have you completed your
objective or 'Goal' during the current cycle?
\033[1m{YellowColor}(y/n) {ResetColor}").lower()
        if stat == "yes" or stat == "y":
            stat = 1
        else:
            stat = 0
        print()
        aflag = True
        while aflag:
            best = input(f"{MainColor} How much is your 'Best
Streak' record of all time in cycles for this habit?
{ResetColor}")
            if best.isdigit():
                aflag = False
            else:
                print(f"{MainColor} Please enter a number!
{ResetColor}")
        print()
        habits.AddHabit(title,goal,int(timefr),int(streak),stat
,int(best))
```

```
    # Remove a habit
    elif user == "r":
        print(habits.ListofHabits())
        user = input(f"{MainColor} Which habit would you like 
to remove? {ResetColor}")
        habits.RemoveHabit(user)

    # Show the table again
    elif user == "s":
        pass

    # Mark a daily task
    elif user == "c":
        print(habits.ListofHabits())
        user = input(f"{MainColor} For which Habit have you 
completed your objective today? {ResetColor}")
        habits.CheckHabit(user)

    # Exit program
    elif user == "q":
        print(f"{MainColor} Thank you for using Habits Tracker! 
{YellowColor}(^_^) ")
        print(f"{MainColor} Have a great day! {ResetColor}")
        break

    # Handle invalid input
    else:
        print(f"\033[91m Invalid Input! Please use the commands 
above! {ResetColor}")
```