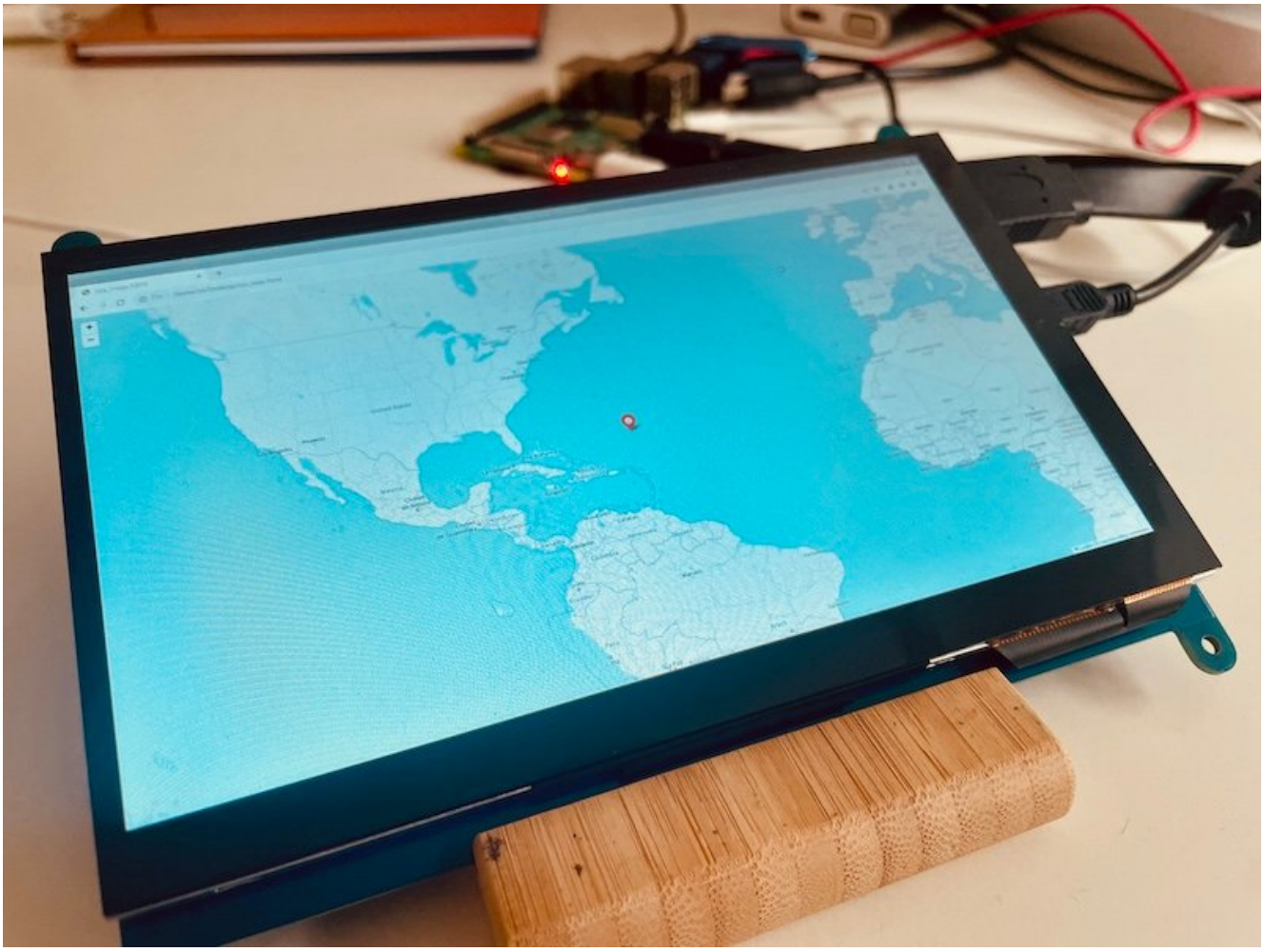
Zeige die Position der ISS auf einer Weltkarte an

Wo befindet sich die ISS (International Space Station) gerade? Diese Frage beantwortest du mit diesem Projekt spielend leicht. Mit Hilfe eines Python-Scripts und einer API **ermittelst du die aktuellen Koordinaten der Raumstation und legst sie über eine Weltkarte.** Diese Karte wird in einem Browser geöffnet und alle 10 Sekunden aktualisiert. So kannst du die aktuelle Position der ISS verfolgen und sehen, über welchem Ort sie sich gerade befindet.

Zum Einsatz kommt in diesem Projekt ein Raspberry Pi. Das Browser-Fenster wird darauf im sogenannten Kiosk-Modus (also im Vollbild) geöffnet. So kannst du mit einem kleinen Display deine eigene ISS-Installation aufbauen.



Der Aufbau des Projekts

Im Prinzip benötigst du nur Python für dieses Projekt. Du kannst das untenstehende Script auch einfach auf deinem PC oder Mac ausführen. Ich lasse es jedoch auf einem Raspberry Pi laufen und verwende statt eines großen Monitors ein kleines 7“ Touch-Display.

[ELECROW Raspberry Pi Display, 7 Zoll Mini Monitor 1024X600 HD LCD Display Touchscreen Monitor Kompatibel mit Raspberry Pi 5/4/3/2/Zero, PC, Jetson Nano, Banana Pi, BB Schwarz]
40,36 €

Mit diesem Script ermittelst du die Position der ISS

Das folgende Python-Script funktioniert so: Von einer API beziehst du die aktuellen Koordinaten, über denen die ISS gerade fliegt. Anschließend erstellst du mit Hilfe der Bibliothek Folium eine Weltkarte und ein Icon, dass die Position der Raumstation anzeigt. Diese Karte speicherst du als HTML-Datei und öffnest diese wiederum im Chrome-Browser. Diese Webseite (also die Weltkarte) wird alle 10 Sekunden aktualisiert, abgespeichert und im Browser aktualisiert.

Installiere die nötigen Bibliotheken

Zunächst zu den Python-Bibliotheken, die du für dieses Projekt benötigst. Installiere diese im Terminal mit den folgenden Befehlen:

```
pip install requests
pip install folium
pip install Selenium
sudo apt-get install chromium-chromedriver
```

Das vollständige Script

Nach der Installation der Bibliotheken, kannst du das folgende Script ausführen. Wie es funktioniert, schauen wir uns gleich an.

```
import requests
import folium
from datetime import datetime
import time
from selenium import webdriver
from selenium.webdriver.chrome.options import Options


def get_iss_position():
                                            response          =
requests.get("http://api.open-notify.org/iss-now.json")
    data = response.json()
    if response.status_code == 200:
                                                timestamp     =
datetime.utcfromtimestamp(data['timestamp']).strftime('%Y-%m-
%d %H:%M:%S')
        latitude = float(data['iss_position']['latitude'])
        longitude = float(data['iss_position']['longitude'])
        return timestamp, latitude, longitude
    else:
        return None
```

```
def main():
    chrome_options = Options()
    chrome_options.add_argument('--kiosk')
      service = webdriver.ChromeService(executable_path =
'/usr/bin/chromedriver')
        driver  =  webdriver.Chrome(service=service,
options=chrome_options)
     driver.get(f"file:///home/pi/Desktop/iss_map.html") #
Passe den Pfad bei Bedarf an
    while True:
        iss_position = get_iss_position()

        if iss_position:
            timestamp, latitude, longitude = iss_position
             print(f"ISS Position at {timestamp}: Latitude
{latitude}, Longitude {longitude}")

                                                iss_map   =
folium.Map(location=[latitude,longitude], zoom_start=4)
            folium.Marker([latitude, longitude], popup=f"ISS
at                                          {timestamp}",
icon=folium.Icon(color='red')).add_to(iss_map)

            # Speichere die Karte als HTML
             iss_map.save("/home/pi/Desktop/iss_map.html") #
Passe den Pfad an deinen eigenen an
            print("Map saved as iss_map.html")
        else:
            print("Failed to retrieve ISS position.")
        time.sleep(10)
        driver.refresh()

if __name__ == "__main__":
    main()
```

So funktioniert das Script

Zunächst importierst du die benötigten Bibliotheken bzw. die notwendigen Funktionen:

```
import requests
import folium
from datetime import datetime
import time
from selenium import webdriver
from selenium.webdriver.chrome.options import Options
```

Dann folgt eine Funktion, mit der du die aktuelle Position der ISS ermittelst:

```
def get_iss_position():
                                        response          =
requests.get("http://api.open-notify.org/iss-now.json")
    data = response.json()
    if response.status_code == 200:
                                            timestamp     =
datetime.utcfromtimestamp(data['timestamp']).strftime('%Y-%m-
%d %H:%M:%S')
        latitude = float(data['iss_position']['latitude'])
        longitude = float(data['iss_position']['longitude'])
        return timestamp, latitude, longitude
    else:
        return None
```

Hier kommt die API von open-notify.org ins Spiel. Diese liefert dir nach ihrem Aufruf die aktuellen Koordinaten der ISS als JSON. Diese Koordinaten sowie die aktuelle Uhrzeit speicherst du in den Variablen **latitude, longitude** und **timestamp**.

In der folgenden Funktion **main()** konfigurierst du zunächst Selenium, die Bibliothek, mit der du die Weltkarte im Browser aufrufst. In der letzten der folgenden Zeilen steckt der Pfad zur HTML-Datei, die die Weltkarte enthält – hierzu gleich mehr.

```
chrome_options = Options()
chrome_options.add_argument('--kiosk')
service  =  webdriver.ChromeService(executable_path  =
```

```
'/usr/bin/chromedriver')
driver      =      webdriver.Chrome(service=service,
options=chrome_options)
driver.get(f"file:///home/pi/Desktop/iss_map.html")  #  Passe
den Pfad bei Bedarf an
```

Zunächst zum Loop, der dafür sorgt, dass du alle 10 Sekunden die aktuelle Position der ISS erhältst.

```
while True:
    iss_position = get_iss_position()

    if iss_position:
        timestamp, latitude, longitude = iss_position
          print(f"ISS Position at {timestamp}: Latitude
{latitude}, Longitude {longitude}")

         iss_map = folium.Map(location=[latitude,longitude],
zoom_start=4)
         folium.Marker([latitude, longitude], popup=f"ISS at
{timestamp}", icon=folium.Icon(color='red')).add_to(iss_map)

        # Speichere die Karte als HTML
         iss_map.save("/home/pi/Desktop/iss_map.html") # Passe
den Pfad an deinen eigenen an
        print("Map saved as iss_map.html")
```

Hier rufst du zunächst die Funktion **get_iss_position()** auf, um die aktuelle Position zu ermitteln. Falls du von dieser Funktion Werte zurück erhältst (der Aufruf der API also geklappt hat), gibst du diese mit **print** aus.

Wichtiger ist jedoch die Weltkarte, die du gleich darauf mit Hilfe der Bibliothek Folium zeichnest. Lass uns auf diese Zeile kurz genauer schauen:

```
iss_map   =   folium.Map(location=[latitude,longitude],
zoom_start=4)
```

Die Parameter **location=[latitude,longitude]** sorgen dafür, dass die Position der ISS auf der Weltkarte zentral angezeigt wird. Die Nadel, die die Raumstation symbolisiert ist also immer im Zentrum zu sehen. Die Welt dreht sich dann quasi unter der ISS weg. Du kannst auch die Einstellung **location=[0,0]** verwenden – dann bleibt die Welt stehen und die ISS-Nadel bewegt sich.

Das führt uns gleich zum Zoom-Level. Im Script oben ist dieser auf 4 eingestellt – ein mittlerer Detailgrad. Je kleiner der Wert, desto mehr zoomst du von der Erde weg. Um genau zu sehen, worüber die ISS gerade fliegt, erhöhe die Zahl in dieser Einstellung.

Zuletzt speicherst du die Karte als HTML-Datei ab. Wenn du hier einen anderen Pfad als den obigen angibst, achte darauf, dass du diesen auch an der Stelle weiter oben im Script entsprechend anpasst.

Damit wären wir fast durch – fehlt nur noch die Angabe, wie oft die Karte aktualisiert werden soll und der Befehl zum Refresh des Browsers:

```
time.sleep(10)
driver.refresh()
```

Hier sind 10 Sekunden eingestellt. Du kannst diesen Wert natürlich beliebig anpassen – beachte jedoch, dass eine zu häufige Aktualisierung zu Problemen mit dem Browser und auch zu einer vorübergehenden Sperrung deiner API-Aufrufe führen kann.

Wie geht es weiter?

Du könntest als Nächstes den Weg der ISS nachzeichnen – also die vergangenen Positionen der ISS auf die Karte übertragen und so den Flug der Raumstation also geschwungene Linie

zeigen. Auch hier kann dir die Bibliothek Folium helfen.