Step 1 - Scraping

NASA Mars News

- Scrape the NASA Mars News Site (https://mars.nasa.gov/news/) and collect the latest News Title and Paragraph Text. Assign the text to variables that you can reference later.

Example:

news_title = "NASA's Next Mars Mission to Investigate Interior of Red Planet"

news_p = "Preparation of NASA's next spacecraft to Mars, InSight, has ramped up this summer, on course for launch next May from Vandenberg Air Force Base in central California -- the first interplanetary launch in history from America's West Coast."

In [1]:
```
# Dependencies
from bs4 import BeautifulSoup as bs
import requests
from splinter import Browser
from datetime import date
from datetime import timedelta
import time
```

In [2]:
```
curr_date = date.today()
y_date = curr_date - timedelta(days = 1)
dby_date = curr_date - timedelta(days = 2)
# print(curr_date, y_date, dby_date)
cstr = curr_date.strftime('%b %d, %Y')
ystr = y_date.strftime('%b %d, %Y')
dstr = dby_date.strftime('%b %d, %Y')
# print(cstr, ystr, dstr)
```

In [3]:
```
# setting up url and opening controlled browser for web scraping
url = 'https://mars.nasa.gov/news/'
executable_path = {'executable_path': '/usr/local/bin/chromedriver'}
browser = Browser('chrome', **executable_path, headless=False)
browser.visit(url)
time.sleep(3)
html = browser.html
soup = bs(html, 'html.parser')
browser.quit()
```

In [4]:
```
res = soup.find_all('li', class_="slide")
```

In [5]:
```
for results in res:
    #print(results.a.h3.text)
    news_title = results.a.h3.text
    #print("------")
    #print(results.a.text)
    news_p = results.a.text
    break
```

```
In [6]: print(news_title)
        print(news_p)
```

```
Air Deliveries Bring NASA's Perseverance Mars Rover Closer to Launch
A NASA Wallops Flight Facility cargo plane transported more than two tons of equ
ipment — including the rover's sample collection tubes — to Florida for this sum
mer's liftoff.Air Deliveries Bring NASA's Perseverance Mars Rover Closer to Laun
ch
```

Part-2

JPL Mars Space Images - Featured Image

- Visit the url for JPL Featured Space Image here (https://www.jpl.nasa.gov/spaceimages/?search=&category=Mars).
- Use splinter to navigate the site and find the image url for the current Featured Mars Image and assign the url string to a variable called `featured_image_url`.
- Make sure to find the image url to the full size `.jpg` image.
- Make sure to save a complete url string for this image.

Example:

featured_image_url = 'https://www.jpl.nasa.gov/spaceimages/images/largesize/PIA16225_hires.jpg (https://www.jpl.nasa.gov/spaceimages/images/largesize/PIA16225_hires.jpg)'

```
In [7]: !which chromedriver
```

```
/usr/local/bin/chromedriver
```

```
In [8]: # Opening the empty headless browser controlled by this code
        executable_path = {'executable_path': '/usr/local/bin/chromedriver'}
        browser = Browser('chrome', **executable_path, headless=False)
```

```
In [9]: url2 = 'https://www.jpl.nasa.gov/spaceimages/?search=&category=Mars'
        browser.visit(url2)
        html = browser.html
        soup2 = bs(html, 'html.parser')
        # Close the browser after scraping
        time.sleep(3)
        browser.quit()
```

```
In [10]: # finding background image title
         title = soup2.find('h1', class_="media_feature_title").text.strip()
         print(title)
```

```
Curiosity Self-Portrait at Martian Sand Dune
```

```
In [11]: # searching title to find the article tag to get the url
         x = 'article'
         res2 = soup2.find(x, alt=title)
         imgurl = res2["style"].split()[1]
         print(imgurl)
```

```
url('/spaceimages/images/wallpaper/PIA20316-1920x1200.jpg');
```

```
In [12]: # spliting unwanted text
         url = imgurl[imgurl.find("'")+1 : imgurl.find(")")-1]
         print(url)
```

```
/spaceimages/images/wallpaper/PIA20316-1920x1200.jpg
```

```
In [13]: #merging image url with website url
         featured_image_url = 'https://jpl.nasa.gov'+url
         print(featured_image_url)
```

```
https://jpl.nasa.gov/spaceimages/images/wallpaper/PIA20316-1920x1200.jpg
```

```
In [14]: # calling the url based on image
         #
         executable_path = {'executable_path': '/usr/local/bin/chromedriver'}
         browser = Browser('chrome', **executable_path, headless=False)
         browser.visit(featured_image_url)
         time.sleep(3)
         browser.quit()
```

Part-3

Mars Weather

- Visit the Mars Weather twitter account here (https://twitter.com/marswxreport?lang=en) and scrape the latest Mars weather tweet from the page. Save the tweet text for the weather report as a variable called `mars_weather` .
- **Note: Be sure you are not signed in to twitter, or scraping may become more difficult.**
- **Note: Twitter frequently changes how information is presented on their website. If you are having difficulty getting the correct html tag data, consider researching Regular Expression Patterns and how they can be used in combination with the .find() method.** # Example: mars_weather = 'Sol 1801 (Aug 30, 2017), Sunny, high -21C/-5F, low -80C/-112F, pressure at 8.82 hPa, daylight 06:09-17:55'

```
In [15]: url3 = 'https://twitter.com/marswxreport?lang=en'
```

```
In [16]: # Retrieve page with the requests module
         response3 = requests.get(url3)
```

```
In [17]: # Create BeautifulSoup object; parse with 'html.parser'
         soup3 = bs(response3.text, 'html.parser')
```

In [18]:
```
res3 = soup3.find_all('div', class_="js-tweet-text-container")
#[aria-label="Timeline: Mars Weather's Tweets"]
for result in res3:
    print(result.p.text)
    print("----")
    if ((result.p.text).find(str(curr_date)) != -1):
        wstr = result.p.text
        wstr = wstr.replace(str(curr_date), cstr)
        break
    elif ((result.p.text).find(str(y_date)) != -1):
        wstr = result.p.text
        wstr = wstr.replace(str(y_date), ystr)
        break
    elif ((result.p.text).find(str(dby_date)) != -1):
        wstr = result.p.text
        wstr = wstr.replace(str(dby_date), dstr)
        break
if len(wstr) > 10:
    wstr = wstr.rsplit(' ',1)[0]
    mars_weather = wstr.split(' ',1)[1]
```

```
InSight sol 530 (2020-05-23) low -92.6ºC (-134.7ºF) high 0.4ºC (32.7ºF)
winds from the SW at 4.7 m/s (10.6 mph) gusting to 17.4 m/s (38.9 mph)
pressure at 7.10 hPapic.twitter.com/LTDrWDwDYH
----
```

In [19]:
```
# print final weather
print(mars_weather)
```

```
sol 530 (May 23, 2020) low -92.6ºC (-134.7ºF) high 0.4ºC (32.7ºF)
winds from the SW at 4.7 m/s (10.6 mph) gusting to 17.4 m/s (38.9 mph)
pressure at 7.10
```

Part-4

Mars Facts

- Visit the Mars Facts webpage here (https://space-facts.com/mars/) and use Pandas to scrape the table containing facts about the planet including Diameter, Mass, etc.
- Use Pandas to convert the data to a HTML table string.

In [20]:
```
#use Pandas for scraping
import pandas as pd
```

In [21]:
```
url4 = 'https://space-facts.com/mars'
tables = pd.read_html(url4)
# tables
```

```
In [22]: df = tables[0]
         df.columns = ['Description','Value']
         df.head(10)
```

Out[22]:

	Description	Value
0	Equatorial Diameter:	6,792 km
1	Polar Diameter:	6,752 km
2	Mass:	6.39 × 10^23 kg (0.11 Earths)
3	Moons:	2 (Phobos & Deimos)
4	Orbit Distance:	227,943,824 km (1.38 AU)
5	Orbit Period:	687 days (1.9 years)
6	Surface Temperature:	-87 to -5 °C
7	First Record:	2nd millennium BC
8	Recorded By:	Egyptian astronomers

```
In [23]: df.set_index('Description', inplace=True)
         df.head(10)
```

Out[23]:

	Value
Description	
Equatorial Diameter:	6,792 km
Polar Diameter:	6,752 km
Mass:	6.39 × 10^23 kg (0.11 Earths)
Moons:	2 (Phobos & Deimos)
Orbit Distance:	227,943,824 km (1.38 AU)
Orbit Period:	687 days (1.9 years)
Surface Temperature:	-87 to -5 °C
First Record:	2nd millennium BC
Recorded By:	Egyptian astronomers

```
In [24]: html_table1 = df.to_html()
         html_table2 = html_table1.replace('border="1"', '')
         html_table = html_table2.replace('dataframe', 'table table-bordered')
         html_table.replace('\n', '')
```

```
Out[24]: '<table  class="table table-bordered">  <thead>    <tr style="text-align: righ
         t;">      <th></th>      <th>Value</th>    </tr>    <tr>      <th>Description</t
         h>      <th></th>    </tr>  </thead>  <tbody>    <tr>      <th>Equatorial Diamet
         er:</th>      <td>6,792 km</td>    </tr>    <tr>      <th>Polar Diameter:</th>
         <td>6,752 km</td>    </tr>    <tr>      <th>Mass:</th>      <td>6.39 × 10^23 kg
         (0.11 Earths)</td>    </tr>    <tr>      <th>Moons:</th>      <td>2 (Phobos &am
         p; Deimos)</td>    </tr>    <tr>      <th>Orbit Distance:</th>      <td>227,943,
         824 km (1.38 AU)</td>    </tr>    <tr>      <th>Orbit Period:</th>      <td>687
         days (1.9 years)</td>    </tr>    <tr>      <th>Surface Temperature:</th>      <
         td>-87 to -5 °C</td>    </tr>    <tr>      <th>First Record:</th>      <td>2nd m
         illennium BC</td>    </tr>    <tr>      <th>Recorded By:</th>      <td>Egyptian
         astronomers</td>    </tr>  </tbody></table>'
```

```
In [25]: import pymongo
```

```
In [26]: conn = 'mongodb://localhost:27017'
         client = pymongo.MongoClient(conn)
         db = client.mars_db
```

```
In [27]: htable = db.marsdata.find()
In [28]: for row in htable:
             print(row['facts'])
```

```
<table  class="table table-bordered table-sm">
  <thead>
    <tr style="text-align: right;">
      <th></th>
      <th>Value</th>
    </tr>
    <tr>
      <th>Description</th>
      <th></th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th>Equatorial Diameter:</th>
      <td>6,792 km</td>
    </tr>
    <tr>
      <th>Polar Diameter:</th>
      <td>6,752 km</td>
    </tr>
    <tr>
      <th>Mass:</th>
      <td>6.39 × 10^23 kg (0.11 Earths)</td>
    </tr>
    <tr>
      <th>Moons:</th>
      <td>2 (Phobos &amp; Deimos)</td>
    </tr>
    <tr>
      <th>Orbit Distance:</th>
      <td>227,943,824 km (1.38 AU)</td>
    </tr>
    <tr>
      <th>Orbit Period:</th>
      <td>687 days (1.9 years)</td>
    </tr>
    <tr>
      <th>Surface Temperature:</th>
      <td>-87 to -5 °C</td>
    </tr>
    <tr>
      <th>First Record:</th>
      <td>2nd millennium BC</td>
    </tr>
    <tr>
      <th>Recorded By:</th>
      <td>Egyptian astronomers</td>
    </tr>
  </tbody>
</table>
```

Part-5

Mars Hemispheres

- Visit the USGS Astrogeology site here (https://astrogeology.usgs.gov/search/results?q=hemisphere+enhanced&k1=target&v1=Mars) to obtain high resolution images for each of Mar's hemispheres.
- You will need to click each of the links to the hemispheres in order to find the image url to the full resolution image.
- Save both the image url string for the full resolution hemisphere image, and the Hemisphere title containing the hemisphere name. Use a Python dictionary to store the data using the keys `img_url` and `title`.
- Append the dictionary with the image url string and the hemisphere title to a list. This list will contain one dictionary for each hemisphere.

Example:

hemisphere_image_urls = [{"title": "Valles Marineris Hemisphere", "img_url": "..."}, {"title": "Cerberus Hemisphere", "img_url": "..."}, {"title": "Schiaparelli Hemisphere", "img_url": "..."}, {"title": "Syrtis Major Hemisphere", "img_url": "..."},]

In [29]:
```
url5 = 'https://astrogeology.usgs.gov/search/results?q=hemisphere+enhanced&k1=tar
get&v1=Mars'
```

In [30]:
```
# Retrieve page with the requests module
response5 = requests.get(url5)
# Create BeautifulSoup object; parse with 'html.parser'
soup5 = bs(response5.text, 'html.parser')
```

In [31]:
```
res = soup5.find_all('div', class_="item")
```

In [32]:
```
# sometimes the website goes down for maintenance
# so checking whether the website is down before scraping....
#
down_mes = '503 Service Temporarily Unavailable'
part5_down = ''
title = []
ref = []
# t1 = soup5.title.text.find(down_mes)
# print(type(t1))
if (soup5.title.text.find(down_mes) != '-1'):
    for x in res:
        try:
            if (x.h3.text and x.a['href']):
                print(x.h3.text)
                title.append(x.h3.text)
                print(x.a['href'])
                ref.append(x.a['href'])
        except AttributeError as e:
            print(e)
else:
    print("Service temporarily unavailable")
    part5_down = down_mes + ' for '+ url5 + ' Please try later.'

# checking whether the list is empty
if not title:
    if len(part5_down) > 5:
        print(part5_down)
    else:
        print("Something went wrong with scraping method")
        part5_down = 'Something went wrong, please notify webadmin.'
else:
    print("-------")
    print(title, ref)
```

```
Cerberus Hemisphere Enhanced
/search/map/Mars/Viking/cerberus_enhanced
Schiaparelli Hemisphere Enhanced
/search/map/Mars/Viking/schiaparelli_enhanced
Syrtis Major Hemisphere Enhanced
/search/map/Mars/Viking/syrtis_major_enhanced
Valles Marineris Hemisphere Enhanced
/search/map/Mars/Viking/valles_marineris_enhanced
-------
['Cerberus Hemisphere Enhanced', 'Schiaparelli Hemisphere Enhanced', 'Syrtis Maj
or Hemisphere Enhanced', 'Valles Marineris Hemisphere Enhanced'] ['/search/map/M
ars/Viking/cerberus_enhanced', '/search/map/Mars/Viking/schiaparelli_enhanced',
'/search/map/Mars/Viking/syrtis_major_enhanced', '/search/map/Mars/Viking/valles
_marineris_enhanced']
```

In [33]:
```
#   source: https://splinter.readthedocs.io/en/latest/elements-in-the-page.html
#    browser.click_link_by_href('http://www.the_site.com/my_link')
#
```

In [34]:
```
# executable_path = {'executable_path': '/usr/local/bin/chromedriver'}
# browser = Browser('chrome', **executable_path, headless=False)
# browser.visit(url5)
# ******************************
# ElementClickInterceptedException: Message: element click intercepted: Element <
a href="/search/map/Mars/Viking/cerberus_enhanced" class="itemLink product-ite
m">...</a> is not clickable at point (122, 217). Other element would receive the
 click: <div class="description">...</div>
#   (Session info: chrome=81.0.4044.138)
# *****************************************
```

In [35]:
```
hemisphere_image_urls = []
if not part5_down:
    for i in range(len(ref)):
        print(ref[i])
        baseurl = 'https://astrogeology.usgs.gov'
        curl = baseurl+ref[i]
        print(curl)
        #browser.click_link_by_href(curl)
        #browser.click_link_by_href(ref[i])
        url51 = curl
        response51 = requests.get(url51)
        soup51 = bs(response51.text, 'html.parser')
        #res51 = soup51.find_all('div', class_="downloads")
        res51 = soup51.find_all('img', class_="wide-image")
        iurl.append(baseurl+res51[0]['src'])
        hemisphere_image_urls.append({"title":title[i], "img_url":baseurl+res51[0
]['src']})
    print(hemisphere_image_urls)

else:
    print(part5_down)
    hemisphere_image_urls.append(part5_down)
```

```
/search/map/Mars/Viking/cerberus_enhanced
https://astrogeology.usgs.gov/search/map/Mars/Viking/cerberus_enhanced

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-35-d6c92472abee> in <module>
     13         #res51 = soup51.find_all('div', class_="downloads")
     14         res51 = soup51.find_all('img', class_="wide-image")
---> 15         iurl.append(baseurl+res51[0]['src'])
     16         hemisphere_image_urls.append({"title":title[i], "img_url":baseur
l+res51[0]['src']})
     17     print(hemisphere_image_urls)

NameError: name 'iurl' is not defined
```

In [ ]:
```
print(hemisphere_image_urls)
```

Merging all output into one single dictionary

In [ ]:
```
print(news_title)
print(news_p)
print(featured_image_url)
print(mars_weather)
print(hemisphere_image_urls)
```

In [ ]:
```
mars_data = {}
mars_data = {
    "news_title":news_title,
    "news_para":news_p,
    "image_url":featured_image_url,
    "weather":mars_weather,
    "facts":html_table,
    "hemisphere_images":hemisphere_image_urls
            }
```

In [ ]:
```
# Module used to connect Python with MongoDb
import pymongo
```

In [ ]:
```
# The default port used by MongoDB is 27017
conn = 'mongodb://localhost:27017'
client = pymongo.MongoClient(conn)

# Define the 'marsDB' database in Mongo
db = client.mars_db
```

In [ ]:
```
# Declare the collection
collection = db.marsdata
```

In [ ]:
```
# remove previous documents before insert
collection.delete_many({})
```

In [ ]:
```
# Insert the document into the database
collection.insert_one(mars_data)
```

In [ ]:
```
# Verify results
results = collection.find()
for result in results:
    print(result)
```

In [ ]:

In [ ]: