**Mission to Mars**

In this assignment, you will build a web application that scrapes various websites for data related to the Mission to Mars and displays the information in a single HTML page. The following outlines what you need to do.

**Step 1 - Scraping**

Complete your initial scraping using Jupyter Notebook, BeautifulSoup, Pandas, and Requests/Splinter.

* Create a Jupyter Notebook file called mission\_to\_mars.ipynb and use this to complete all of your scraping and analysis tasks. The following outlines what you need to scrape.

**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."

**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'

**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.

# 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'

**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.

**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": "..."},

]

**Step 2 - MongoDB and Flask Application**

Use MongoDB with Flask templating to create a new HTML page that displays all of the information that was scraped from the URLs above.

* Start by converting your Jupyter notebook into a Python script called scrape\_mars.py with a function called scrapethat will execute all of your scraping code from above and return one Python dictionary containing all of the scraped data.
* Next, create a route called /scrape that will import your scrape\_mars.py script and call your scrape function.
  + Store the return value in Mongo as a Python dictionary.
* Create a root route / that will query your Mongo database and pass the mars data into an HTML template to display the data.
* Create a template HTML file called index.html that will take the mars data dictionary and display all of the data in the appropriate HTML elements. Use the following as a guide for what the final product should look like, but feel free to create your own design.

**Step 3 - Submission**

To submit your work to BootCampSpot, create a new GitHub repository and upload the following:

1. The Jupyter Notebook containing the scraping code used.
2. Screenshots of your final application.
3. Submit the link to your new repository to BootCampSpot.

**Hints**

* Use Splinter to navigate the sites when needed and BeautifulSoup to help find and parse out the necessary data.
* Use Pymongo for CRUD applications for your database. For this homework, you can simply overwrite the existing document each time the /scrape url is visited and new data is obtained.
* Use Bootstrap to structure your HTML template.

**Copyright**

© 2019 Trilogy Education Services. All Rights Reserved.