10.3.6 Export to Python

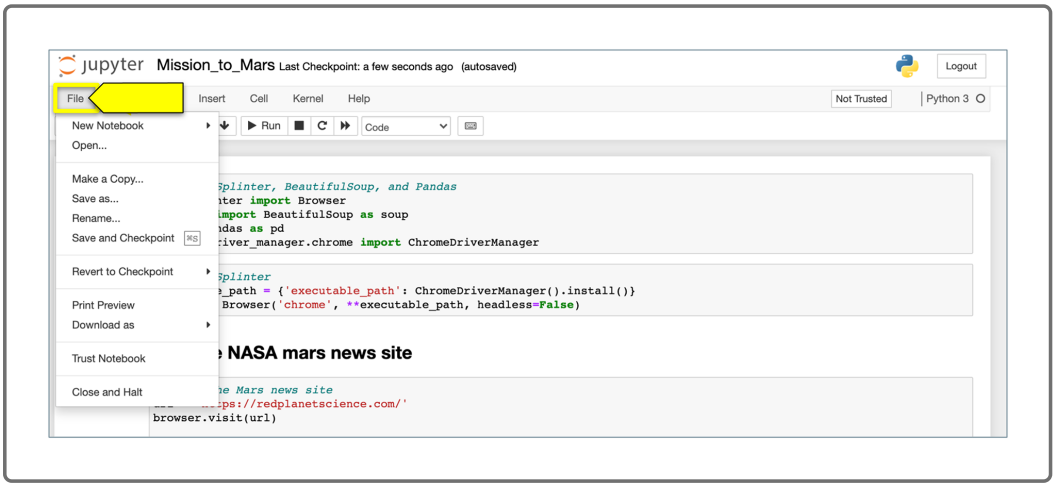
All of the scraping work is coming together. Robin's code can pull article summaries and titles, a table of facts, and a featured image. This is awesome. And Jupyter Notebook is the perfect tool for building a scraping script. We can build it in chunks: one chunk for the image, one chunk for the article, and another for the facts. Each chunk can be tested and run independently from the others. However, we can't automate the scraping using the Jupyter Notebook. To fully automate it, it will need to be converted into a `.py` file.

The next step in making this an automated process is to download the current code into a Python file. It won't transition over perfectly, we'll need to clean it up a bit, but it's an easier task than copying each cell and pasting it over in the correct order.

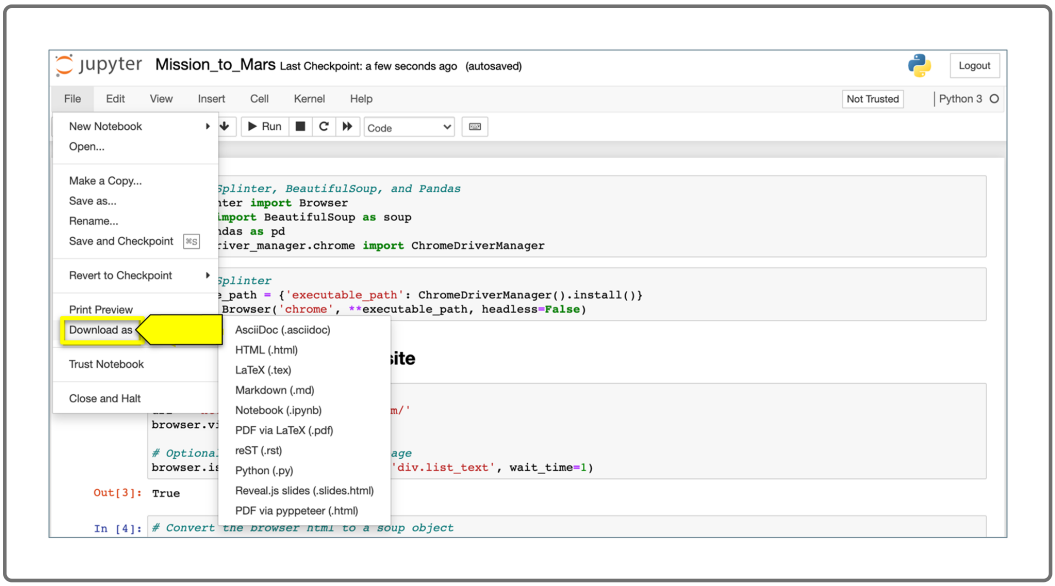
The Jupyter ecosystem is an extremely versatile tool. We already know many of its great functions, such as the different libraries that work well with it and also how easy it is to troubleshoot code. Another feature is being able to download the notebook into different formats.

There are several formats available, but we'll focus on one by downloading to a Python file.

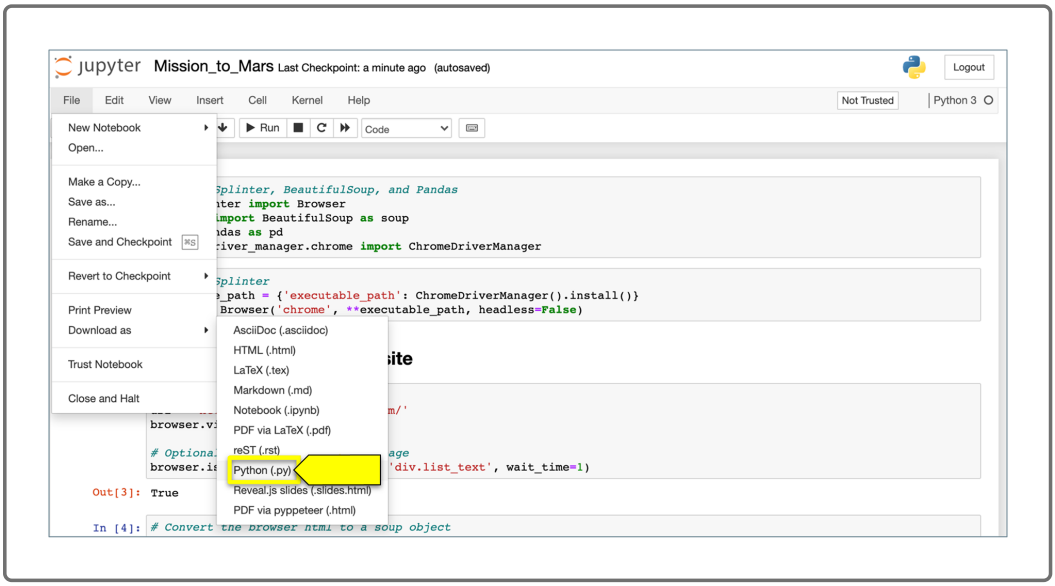
1. While your notebook is open, navigate to the top of the page to the Files tab.



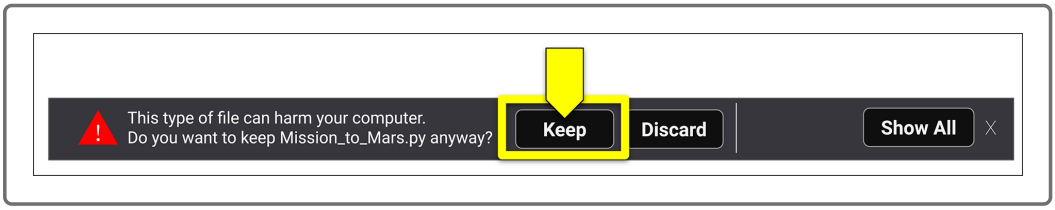
2. From here, scroll down to the "Download as" section of the drop-down menu.



3. Select "Python (.py)" from the next menu to download the code.



4. If you get a warning about downloading this type of file, click "Keep" to continue the download.



5. Navigate to your Downloads folder and open the new file. A brief look at the first lines of code shows us that the code wasn't the only thing to be ported over. The number of times each cell has been run is also there, for example.

```
#!/usr/bin/env python
# coding: utf-8

Run Cell | Run Below | Debug Cell
# In[1]:


# Import Splinter, BeautifulSoup, and Pandas
from splinter import Browser
from bs4 import BeautifulSoup as soup
import pandas as pd
from webdriver_manager.chrome import ChromeDriverManager


Run Cell | Run Above | Debug Cell
# In[ ]:


# Set up Splinter
executable_path = {'executable_path': ChromeDriverManager().install()}
browser = Browser('chrome', **executable_path, headless=False)


# ## Visit the NASA mars news site

Run Cell | Run Above | Debug Cell
# In[3]:


# Visit the Mars news site
url = 'https://redplanetscience.com/'
browser.visit(url)
```

6. Clean up the code by removing unnecessary blank spaces and comments.

```
# Import Splinter, BeautifulSoup, and Pandas
from splinter import Browser
from bs4 import BeautifulSoup as soup
import pandas as pd
from webdriver_manager.chrome import ChromeDriverManager

# Set up Splinter
executable_path = {'executable_path': ChromeDriverManager().install()}
browser = Browser('chrome', **executable_path, headless=False)


# Visit the Mars news site
url = 'https://redplanetscience.com/'
browser.visit(url)

# Optional delay for loading the page
browser.is_element_present_by_css('div.list_text', wait_time=1)

# Convert the browser html to a soup object
html = browser.html
news_soup = soup(html, 'html.parser')

slide_elem = news_soup.select_one('div.list_text')
slide_elem.find('div', class_='content_title')

# Use the parent element to find the first a tag and save it as `news_title`
news_title = slide_elem.find('div', class_='content_title').get_text()
news_title

# Use the parent element to find the paragraph text
news_p = slide_elem.find('div', class_='article_teaser_body').get_text()
news_p
```

When you're done tidying up the code, make sure you save it in your working folder with your notebook code as scraping.py. You can also test the script by running it through your terminal.

The final scraping.py file should look like this:

```
# find the relative image url
img_url_rel = img_soup.find('img', class_='fancybox-image').get('src')
img_url_rel

# Use the base url to create an absolute url
img_url = f'https://spaceimages-mars.com/{img_url_rel}'
img_url

# ## Mars Facts

df = pd.read_html('https://galaxyfacts-mars.com')[0]
df.head()

df.columns=['Description', 'Mars', 'Earth']
df.set_index('Description', inplace=True)
df

df.to_html()

browser.quit()
```

Awesome! Although it was a little tedious to clean up the code, at least you didn't have to rewrite everything or copy and paste the code cell by cell. Also, Jupyter Notebook can be converted to other file types as well, such as markdown and text, so this is a really useful skill to have.