

Web Scrapping Stargate Scripts

<https://imsdb.com/TV/Stargate%20SG1.html> (<https://imsdb.com/TV/Stargate%20SG1.html>)

```
In [1]: 1 import requests
2 from bs4 import BeautifulSoup
3 from bs4 import element
4 import re
5 import pandas as pd
6 import pickle
7 import json

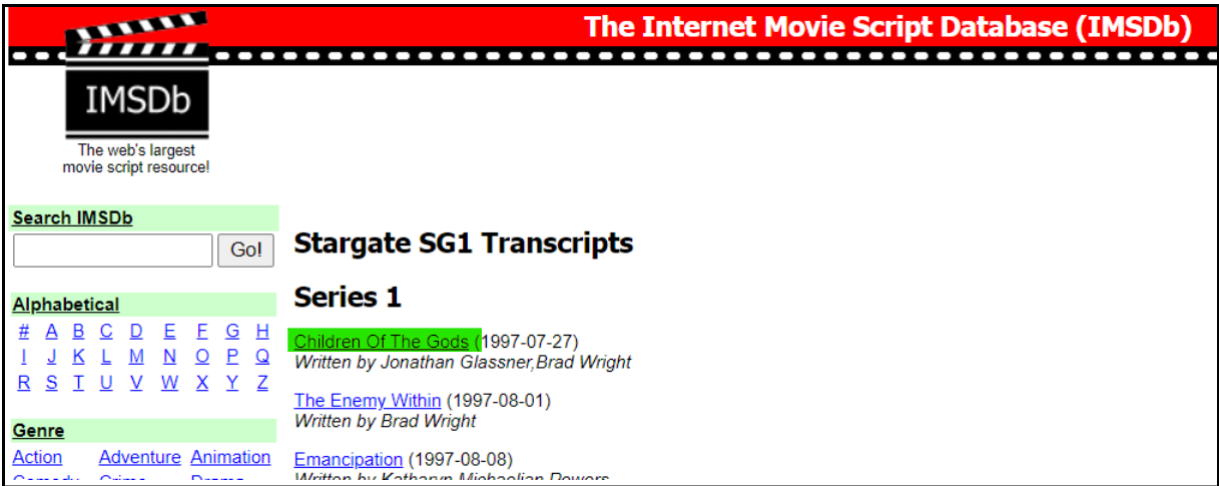
...

In [2]: 1 pd.set_option('display.max_columns', None) # or 1000
2 pd.set_option('display.max_rows', None) # or 1000
3 pd.set_option('display.max_colwidth', None) # or 199
```

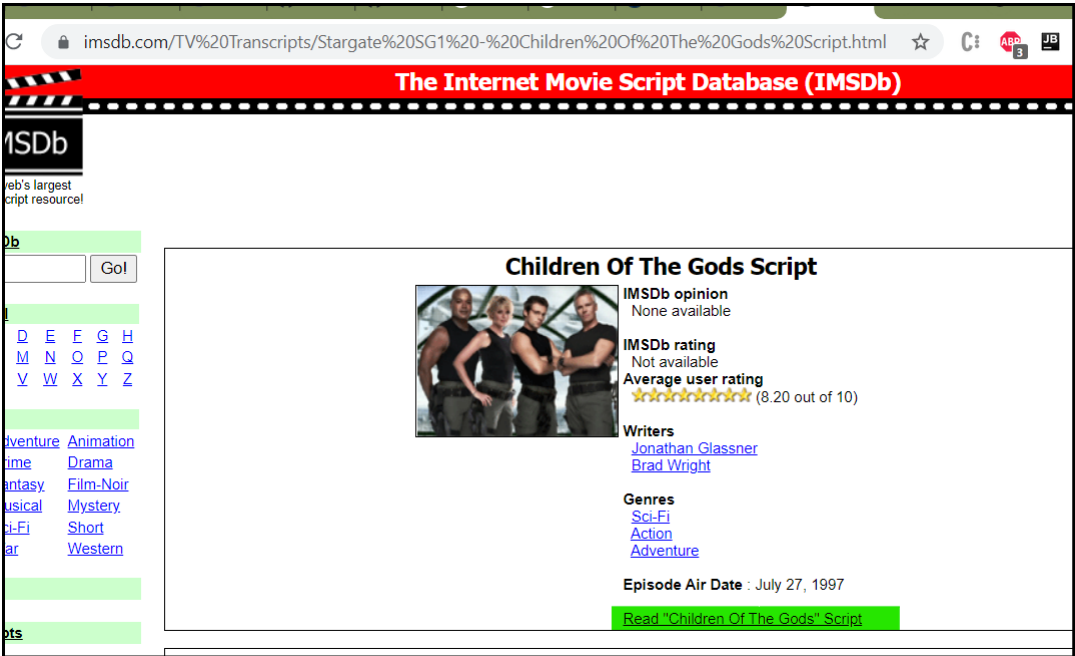
Obtaining the Links to the Scripts

<https://imsdb.com/TV/Stargate%20SG1.html> (<https://imsdb.com/TV/Stargate%20SG1.html>)

1) Obtain all episode links



2) Obtain link for script



```
In [3]: 1 # Obtain HTML code of Main Page
2 # source: https://stackoverflow.com/questions/45204152/beautifulsoup-pull-p-tag-while-between-defined-h2-tags
3 URL = "https://imsdb.com/TV/Stargate%20SG1.html"
4 page = requests.get(URL)
5 #page.text
```

```
In [4]: 1 # Parse HTML page with beautiful soup
2 soup = BeautifulSoup(page.content, "html.parser")
3
4 # h2 tags with season name
5 seasons = [t.text for t in soup.find_all("h2")]
6
7 # Extracting the links for all episodes
8 # --List comprehension with python dictionary
9 # --Dictionary will store the episode links sorted by season
10 episode_links = {season_nr:{} for season_nr in range(1,len(seasons)+1,1)}
11 season_count = 1
12 episode_count = 1
13
14 # Populate the episode link dictionary
15 for season_tag in soup.find_all("h2"):
16     episode_count = 1
17     # Get all episodes that follow a "season headline" (see image 1 above)
18     for tag in season_tag.next_siblings:
19         if isinstance(tag, element.Tag):
20
21             # go to next season
22             if season_count < 9 and tag.text == seasons[season_count]: # check if the current tag contains next "season headline"
23                 season_count+=1
24                 break
25
26             # go to next episode
27             if tag.find("a"):
28                 episode_links[season_count][episode_count] = "https://imdb.com" + tag.find("a")['href']
29                 episode_count+=1
30
31 del episode_links[6][14] # episode is missing
32 #episode_links
33
34 >
```

Obtaining the Scripts' HTML → Creating Pandas Dataframes

```
In [5]: 1 def get_script(soup:BeautifulSoup)->element.Tag:
2     """Obtain a script's HTML code"""
3     for tag in soup.find_all('a'):
4         if re.match(r"Read\s\.*\sScript", tag.text) and isinstance(tag, element.Tag):
5             URL = "https://imdb.com"+tag['href']
6             page = requests.get(URL)
7             soup = BeautifulSoup(page.content, 'html.parser')
8             script = soup.find("pre").find("body")
9             return script
10            return None

In [6]: 1 def string_empty(string:str)->bool:
2     """Check if string is empty or empty space characters"""
3     # source: https://thispointer.com/python-check-if-string-is-empty-or-blank-or-contain-spaces-only/
4     return not string.text.strip() or string.text.isspace()
5
6 def is_tag(tag)->bool:
7     return isinstance(tag, element.Tag)
8
9 def is_text(text)->bool:
10    return isinstance(text, element.NavigableString)
11
12 # Very rudimental data cleaning
13 def remove_newline(string:str)->str:
14    return string.replace("\n", "").replace("\r", "")

In [7]: 1 def script_to_df(script: element.Tag):
2     """Extract the df from a html movie script.
3     Puts the content into a two column dataframe with one column marking the speaking character
4     and another column which contains the spoken text."""
5     script_table = []
6     for tag in script.find_all("b"):
7         if is_tag(tag) and string_empty(tag):
8             sibling = tag.next_sibling
9             # get character dialogue
10            if is_tag(sibling) and not string_empty(sibling):
11                script_table.append([remove_newline(str(sibling.text)), remove_newline(str(sibling.next_sibling))])
12            # get intermediate text (all text that is not a character speaking)
13            elif is_text(sibling):
14                script_table.append(["interlude", remove_newline(str(sibling))])
15    script_df = pd.DataFrame(script_table)
16    script_df.columns = ["character", "text"]
17    return script_df

In [8]: 1 # Dictionary that contains the links to the dataframes sorted by season and episode number
2 corpus_dict = {season_nr:{} for season_nr in range(1,len(seasons)+1,1)}
3 #corpus dict

In [9]: 1 raw_data_folder = "../raw data/"

In [10]: 1 # Create data
2 for season_nr, episodes in episode_links.items():
3     for episode_nr, episode_link in episodes.items():
4
5         # Create DataFrame
6         page = requests.get(episode_link)
7         soup = BeautifulSoup(page.content, 'html.parser')
8         script = get_script(soup)
9         script_df = script_to_df(script)
10
11        # Save DataFrame pickle
12        filename = "S"+str(season_nr)+"-E"+str(episode_nr)+".pkl"
13        script_df.to_pickle(raw_data_folder+filename)
14        corpus_dict[season_nr][episode_nr] = filename

In [11]: 1 # pickle corpus (store)
2 # source: https://stackoverflow.com/questions/11218477/how-can-i-use-pickle-to-save-a-dict
3 with open(raw_data_folder+'corpus_dict.pkl', 'wb') as handle:
4     pickle.dump(corpus_dict, handle, protocol=pickle.HIGHEST_PROTOCOL)
```

In [12]:

```
1 # Load data (deserialize)
2 with open(raw_data_folder+'corpus_dict.pkl', 'rb') as handle:
3     corpus_dict = pickle.load(handle)
4
5 json.loads(json.dumps(corpus_dict))
```

Out[12]:

```
{'1': {'1': 'S1-E1.pkl',
'2': 'S1-E2.pkl',
'3': 'S1-E3.pkl',
'4': 'S1-E4.pkl',
'5': 'S1-E5.pkl',
'6': 'S1-E6.pkl',
'7': 'S1-E7.pkl',
'8': 'S1-E8.pkl',
'9': 'S1-E9.pkl',
'10': 'S1-E10.pkl',
'11': 'S1-E11.pkl',
'12': 'S1-E12.pkl',
'13': 'S1-E13.pkl',
'14': 'S1-E14.pkl',
'15': 'S1-E15.pkl',
'16': 'S1-E16.pkl',
'17': 'S1-E17.pkl',
'18': 'S1-E18.pkl',
'19': 'S1-E19.pkl',
'20': 'S1-E20.pkl',
'21': 'S1-E21.pkl',
'22': 'S1-E22.pkl',
'23': 'S1-E23.pkl',
'24': 'S1-E24.pkl',
'25': 'S1-E25.pkl',
'26': 'S1-E26.pkl',
'27': 'S1-E27.pkl',
'28': 'S1-E28.pkl',
'29': 'S1-E29.pkl',
'30': 'S1-E30.pkl',
'31': 'S1-E31.pkl',
'32': 'S1-E32.pkl',
'33': 'S1-E33.pkl',
'34': 'S1-E34.pkl',
'35': 'S1-E35.pkl',
'36': 'S1-E36.pkl',
'37': 'S1-E37.pkl',
'38': 'S1-E38.pkl',
'39': 'S1-E39.pkl',
'40': 'S1-E40.pkl',
'41': 'S1-E41.pkl',
'42': 'S1-E42.pkl',
'43': 'S1-E43.pkl',
'44': 'S1-E44.pkl',
'45': 'S1-E45.pkl',
'46': 'S1-E46.pkl',
'47': 'S1-E47.pkl',
'48': 'S1-E48.pkl',
'49': 'S1-E49.pkl',
'50': 'S1-E50.pkl',
'51': 'S1-E51.pkl',
'52': 'S1-E52.pkl',
'53': 'S1-E53.pkl',
'54': 'S1-E54.pkl',
'55': 'S1-E55.pkl',
'56': 'S1-E56.pkl',
'57': 'S1-E57.pkl',
'58': 'S1-E58.pkl',
'59': 'S1-E59.pkl',
'60': 'S1-E60.pkl',
'61': 'S1-E61.pkl',
'62': 'S1-E62.pkl',
'63': 'S1-E63.pkl',
'64': 'S1-E64.pkl',
'65': 'S1-E65.pkl',
'66': 'S1-E66.pkl',
'67': 'S1-E67.pkl',
'68': 'S1-E68.pkl',
'69': 'S1-E69.pkl',
'70': 'S1-E70.pkl',
'71': 'S1-E71.pkl',
'72': 'S1-E72.pkl',
'73': 'S1-E73.pkl',
'74': 'S1-E74.pkl',
'75': 'S1-E75.pkl',
'76': 'S1-E76.pkl',
'77': 'S1-E77.pkl',
'78': 'S1-E78.pkl',
'79': 'S1-E79.pkl',
'80': 'S1-E80.pkl',
'81': 'S1-E81.pkl',
'82': 'S1-E82.pkl',
'83': 'S1-E83.pkl',
'84': 'S1-E84.pkl',
'85': 'S1-E85.pkl',
'86': 'S1-E86.pkl',
'87': 'S1-E87.pkl',
'88': 'S1-E88.pkl',
'89': 'S1-E89.pkl',
'90': 'S1-E90.pkl',
'91': 'S1-E91.pkl',
'92': 'S1-E92.pkl',
'93': 'S1-E93.pkl',
'94': 'S1-E94.pkl',
'95': 'S1-E95.pkl',
'96': 'S1-E96.pkl',
'97': 'S1-E97.pkl',
'98': 'S1-E98.pkl',
'99': 'S1-E99.pkl',
'100': 'S1-E100.pkl'}}
```

In [13]:

```
1 # Example df
2 with open(raw_data_folder+corpus_dict[2][3], 'rb') as handle:
3     unserialized_data = pickle.load(handle)
4
5 pickle.loads(pickle.dumps(unserialized_data))
```

Out[13]:

	character	text
0	JACK	All right kids. We're due back, unless you can tell me you've discovered something Earth shattering. I'm read to bag this one.
1	DANIEL	We practically just got here. We have no idea what this planet has to offer.
2	JACK	Trees and Moss.
3	DANIEL	Well, a few miles from the Stargate, granted, but...
4	JACK	Captain any signs of radio traffic in the last twenty-four hours? Any chemical traces in the air to indicate civilization?
5	SAM	None, sir.
6	interlude	Daniel tries to speak and puts his finger in the air. Jack points his own finger
7	JACK	Ah! Ah! Well flag it for an aerial survey. As much as I love a good rain forest..
8	TEAL'C	O'Neill
9	interlude	A man comes out of the tree's and falls to his knees
10	MAN	Help me. They find me. Taldor.
11	JACK	Who?