Python to EXE

https://pypi.org/project/auto-py-to-exe/

Downloading web browser drivers

1. Firefox: gecko driver
   https://github.com/mozilla/geckodriver/releases
2. Chrome: chrome driver
   https://chromedriver.chromium.org
3. Edge: edge web driver
   https://developer.microsoft.com/en-us/microsoft-edge/tools/webdriver/

When to use Selenium

We use Selenium when:

- When Requests and BeautifulSoup does not work.
- When page requires JavaScript to render the data.

Pros:

- It launches real browser and automate browser.
- Better compatibility.

Cons:

- Slow because it launches real browser.

Essential Selenium Functions

```
from selenium import webdriver

# Choose either one browser driver
firefox =
webdriver.Firefox(executable_path='drivers\geckodriver.exe')
chrome =
webdriver.Chrome(executable_path='drivers\chromedriver.exe')
edge =
webdriver.Edge(executable_path='drivers\MicrosoftWebDriver.exe')

# Headless mode, Chrome example
from selenium.webdriver.chrome.options import Options
options = Options()
options.add_argument('-headless')
browser = webdriver.Chrome(options=options)

# Maximize window
browser.maximize_window()

# Controlling navigation
browser.get("https://example.com")
browser.back()
browser.forward()
browser.refresh()

# Finding elements by CSS selectors
browser.find_element_by_css_selector("h1")
browser.find_elements_by_css_selector("main a")

# Switch to window or frame
# useful for pop-ups
browser.switch_to.window("window_name")
browser.switch_to.frame(1)
browser.switch_to.frame("frame_name)
browser.switch_to.parent_frame()

# Cookies
# useful for controlling logged in session
cookies_list = browser.get_cookies()
browser.get_cookie("my_cookie")
browser.add_cookie({"name:value"})
browser.delete_cookie("my_cookie")
browser.delete_all_cookies()

# Form input
input_element.send_keys("something from keyboard")

# Form input, pressing enter
from selenium.webdriver.common.keys import Keys
input_element.send_keys(Keys.RETURN)

# Execute JavaScript
```

```
browser.execute_script("alert('hello')")
```

More usage in following cheatsheets:
https://mak.la/selenium

Taking screenshot Selenium basic usage

```
from selenium import webdriver
from selenium.webdriver.chrome.options import Options

options = Options()
options.add_argument('-headless')

browser = webdriver.Chrome(options=options)
browser.maximize_window()
browser.get('http://macaodaily.com')
browser.save_screenshot('MacaoDaily.png')
browser.quit()
```

Code example: Fetch DICJ data

```
from selenium import webdriver
from selenium.webdriver.chrome.options import Options
import time

options = Options()
options.add_argument('-headless')

browser = webdriver.Chrome(options=options)

browser.get('http://www.dicj.gov.mo/web/cn/information/
DadosEstat_mensal/2020/index.html')

time.sleep(5)

element = browser.find_element_by_css_selector("#report
#table1")

rows = element.find_elements_by_css_selector("tr")
print(rows[0].text)
for row in rows[3:]:
    print(row.text)
```

Example of sending email via Mailgun

Note: you will need to put in your API keys.

```
def send_simple_message(content, subject="Yeah"):
    return requests.post(
        f"https://api.mailgun.net/v3/{DOMAIN}/messages",
        auth=("api", API_KEY),
        data={"from": FROM,
        "to": TO,
        "subject": subject,
        "text": content})
```

Discussion: Shall we fetch online or automate the web?

1. Check /robots.txt
   https://www.robotstxt.org

   e.g. https://www.dsat.gov.mo/robots.txt

   ```
   User-agent: *
   Disallow: /dsat_bres/
   Allow: /
   ```

   e.g. https://www.malimalihome.net/robots.txt

2. Scale of automations
   Single thread, a few threads vs. Large-scale, multiple threads
3. Usage purpose
   Self-usage, group-usage, or reselling for profits?