INF344 2019–2020

Description | Submission view

TP de collecte de données

Disponible à partir de: Friday 10 May 2019, 12:56
Due date: Monday 20 May 2019, 00:00
Requested files: collector.py (Download)
Nombre maximal de fichiers: 20
Type of work: Individual work

Objectif: acquérir une pratique de méthodes de collecte de données sur le Web

Pour ce TP, vous allez utiliser diverses techniques de collecte de données sur le Web: parcours de pages web, analyse de pages, extraction d'information. ce TP utilise Python.

Nous allons principalement nous intéresser au site http://www.freepatentsonline.com. Le but va être d'extraire des informations sur des brevets. Nous allons utiliser divers outils pour réaliser cette collecte de données, du plus simple au plus élaboré.

A la fin du TP, ou, au plus tard le 19/5 à minuit, vous devrez avoir déposé vos fichiers sur le moodle. Les dépôts tardifs seront pénalisés.

ATTENTION:

lors des requêtes répétées à un même site et compte tenu du nombre que vous êtes, respectez un délai d'au moins 2 secondes entre 2 appels: pensez-y dans votre code, par exemple en utilisant la méthode sleep entre 2 requêtes ; le non-respect de cette règle risque d'entrainer le blocage de nos accès

Eléments fournis

- un modèle de code de base nommé collector.py (dans Moodle).

Impératif: vous DEVEZ mettre votre login de Telecom Paris dans collector.py à la place de "YOUR NAME HERE".

Tâches

Modifiez le modèle collector.py en le complétant avec votre code. Il y a une méthode stepx à compléter par étape x décrite ci-après. L'étape step0 sert seulement à valider que vous avez indiqué votre login dans le main.

Etape 1 (notée 1 point)
Commencez par compléter la méthode step1 par du code qui fait la requête

http://www.freepatentsonline.com/result.html?sort=relevance&srch=top&query_txt=video&submit=&patents=on

sur le site FPO pour charger (par exemple en utilisant urllib) et récupérer dans la variable result la réponse du serveur FPO à cette requête. Cette requête interroge le serveur pour obtenir une liste de brevets concernant la vidéo. La fin de step1 assure la sauvegarde du fichier correspondant sous le nom collecte.step1.

Etape 2 (notée sur 3)
Vous allez analyser le contenu précédemment chargé (ou le recharger). Pour commencer, vous pouvez utiliser BeautifulSoup (import bs4) pour corriger les défauts éventuels de la page chargée et analyser la page. Le but est de récupérer les liens (href) contenus dans toutes les balises <a> de la page. Le résultat à sauver dans le fichier est une chaîne de caractère composée de chaque valeur href de chaque balise <a>, une par ligne (pour une liste links de liens, on pourra par exemple générer la chaine à sauver avec le code result = "\n".join(links)

Etape 3 (notée 3 point)
Vous allez compléter la méthode linksfilter. Elle reçoit une liste de liens qu'elle filtre et renvoie la liste de liens filtrée. Le but est:
* d'éviter de retrouver plusieurs fois le même lien,
* d'éliminer certains liens; ici on éliminera: /, /services.html, /contact.html, /privacy.html, /register.html, /tools-resources.html, https://twitter.com/FPOCommunity, http://www.linkedin.com/groups/FPO-Community-4524797, http://www.sumobrainsolutions.com/ et les liens commençant par result.html ou http://research../search.html

4324797, http://www.sumobrainsolutions.com/ et les liens commençant par result.html ou http://research/search.html,

Dans la méthode step3, vous appelerez la méthode linksfilter sur les liens obtenus à l'étape 2, puis vous trierez par ordre alphabétique les liens restants.
Vous sauverez la liste des liens obtenus dans un fichier texte contenant un lien par ligne.

Etape 4 (notée 3 points)

Note: à partir de l'étape 4, le traitement peut prendre un peu de temps; essayez le mettre au point progressivement en limitant les appels au réseau

Vous allez suivre les 10 premiers liens rencontrés à l'étape 3 (pour cette étape en particulier, pensez à établir le délai entre les requêtes, voir remarque plus haut). Dans les pages obtenues, vous allez récolter les liens. Dans le fichier que vous allez générer à cette étape, les lignes sont constituées des liens rencontrés dans ces pages, après dédoublonnage de l'ensemble et tri par ordre alphabétique (méthode sort sur les chaînes).

Etape 5 (notée 2 points)

Ici, vous allez suivre les liens obtenus à l'étape 4. Vous allez ouvrir les pages correspondantes (s'il s'agit de html) et appeler la méthode contentfilter qui renvoie True si la page est considérée comme intéressante et False sinon. Vous allez compléter la méthode contentfilter par du code qui teste si des de la page contiennent les textes "Inventors:", "Title:" et "Application Number:".

Par exemple, pour tester la présence de "Inventors:", vous pouvez utiliser le code suivant:

```
inventors = soup.find_all(text=re.compile('Inventors:'))
```

```
et tester si inventors est différent de None.
```

```
Vous testerez les liens de l'étape 4 jusqu'à avoir obtenu 10 liens. Vous sauverez cette liste de 10 liens (ici encore une ligne par lien dans un fichier texte).
```

```
La liste sera triée par ordre alphabétique avant d'être sauvée.
```

Etape 6 (noté 4 points)
Pour les 5 premiers éléments obtenus dans le résultat de l'étape 5, vous allez constituer la liste des inventeurs que vous sauverez dans le fichier de résultat de cette étape.
Note: on peut s'inspirer du code suivant pour atteindre la liste des inventeurs (voir étape 5 pour récupérer inventors)

```
divs = [inventor.parent.parent for inventor in inventors]
for d in divs[0].descendants:
    if d.name == 'div' and d.get('class', '') == ['disp_elm_text']:
        inventorlist = d.text
```

Le fichier résultat contient un inventeur par ligne (on conservera l'intégralité du texte présent pour chaque inventeur).

Etape 7 (3 points)
Pour cette étape, vous allez créer un fichier README<votre login>.txt dans lequel vous commenterez:
* le rôle général d'une méthode comme linksfilter dans une application de crawling,
* le rôle général d'une méthode comme contentfilter dans une application de crawling et les limites du code proposé à l'étape 6.
Vous sauverez ce fichier en utilisant l'onglet submission. Les points obtenus pour cette étape seront ajoutés aux points obtenus par l'évaluation automatique du code.
Dans ce README, vous pourrez aussi mettre toute remarque que vous jugerez utile.

Code

Ce TP peut être programmé, exécuté et évalué directement dans Moodle.

Dans la tabulation “Edit”, vous trouverez le fichier collector.py que vous devrez compléter. Vous pouvez directement modifier le code dans l'interface web. Cliquez sur l'icone 'disque' pour sauver votre travail. Cliquez l'icone 'fusée' pour exécuter votre code. Cliquez l'icone " pour le déboguer.

Si vous préférez travailler hors ligne, par exemple avec vos outils de développement habituels: vous pouvez télécharger le code, le modifier puis le recharger dans Moodle à l'aide de l'onglet Submission.

Vous pouvez utiliser les librairies suivantes: . Nous ne pouvons garantir le bon fonctionnement avec d'autres librairies non standard.

Soumettre votre TP

Lorsque vous avez terminé, vous devez soumettre votre travail (cliquez la case à cocher). Cela va automatiquement calculer une évaluation estimée à partir de données de référence. Vous pouvez soumettre plusieurs fois sans pénalité; la dernière soumission sera prise en compte.

Vous pouvez vous conseiller les uns et les autres, mais le plagiat est sanctionné, potentiellement avec une note de 0/20 et éventuellement d'autres sanctions.

NOTE: le serveur de soumission et d'évaluation ne fonctionne qu'à l'intérieur Télécom ParisTech. Référez-vous à https://www.telecom-paristech.fr/vivre-ecole/services-numeriques-dsi/connexion-depuis-lexterieur.html pour savoir comment vous connecter depuis l'extérieur de l'école en étant vu comme étant sur le réseau interne (VPN).

Requested files

collector.py

```python
# -*- coding: utf-8 -*-
# écrit par Jean-Claude Moissinac, structure du code par Julien Romero

from sys import argv
import sys
if (sys.version_info > (3, 0)):
    from urllib.request import urlopen
    from urllib.parse import urlencode
else:
    from urllib2 import urlopen
    from urllib import urlencode

class Collecte:
    """pour pratiquer plusieurs méthodes de collecte de données"""

    def __init__(self):
        """__init__
        Initialise la session de collecte
        :return: Object of class Collecte
        """
        # DO NOT MODIFY
        self.basename = "collecte.step"

    def collectes(self):
        """collectes
        Plusieurs étapes de collectes. VOTRE CODE VA VENIR CI-DESSOUS
        COMPLETER les méthodes stepX.
        """
        self.step0()
        self.step1()
        self.step2()
        self.step3()
        self.step4()
        self.step5()
        self.step6()

    def step0(self):
        # cette étape ne sert qu'à valider que tout est en ordre; rien à coder
        stepfilename = self.basename+"0"
        print("Comment :=>> Validation de la configuration")
        with open(stepfilename, "w", encoding="utf-8") as resfile:
            resfile.write(self.name)

    def step1(self):
        stepfilename = self.basename+"1"
        result = ""
        # votre code ici
        with open(stepfilename, "w", encoding="utf-8") as resfile:
            resfile.write(result)

    def step2(self):
        stepfilename = self.basename+"2"
        result = ""
        # votre code ici
        with open(stepfilename, "w", encoding="utf-8") as resfile:
            resfile.write(result)

    def linksfilter(self, links):
        flinks = links
        # votre code ici
        return flinks

    def step3(self):
        stepfilename = self.basename+"3"
        result = ""
        # votre code ici
        with open(stepfilename, "w", encoding="utf-8") as resfile:
            resfile.write(result)

    def step4(self):
        stepfilename = self.basename+"4"
        result = ""
        # votre code ici
        with open(stepfilename, "w", encoding="utf-8") as resfile:
            resfile.write(result)

    def contentfilter(self, link):
        return False

    def step5(self):
        stepfilename = self.basename+"5"
        result = ""
        # votre code ici
        with open(stepfilename, "w", encoding="utf-8") as resfile:
            resfile.write(result)

    def step6(self):
        stepfilename = self.basename+"6"
        result = ""
        # votre code ici
        with open(stepfilename, "w", encoding="utf-8") as resfile:
            resfile.write(result)

if __name__ == "__main__":
    collecte = Collecte()
    collecte.collectes()

```

VPL

◄ Slides Premiere Session Part2

Aller à...

Connecté sous le nom « Romain Legrand » (Déconnexion)
INF344 2019–2020
Résumé de conservation de données
Obtenir l'app mobile