Wikigame

Édition 2024

Un TP de webscraping à écrire en Python 3 avec BeautifullSoup ou Scrapy (ou autre chose).

TP à réaliser seul.

Niveau : Développeurs Python confirmés (Bac + 3 / Bac + 4)

1 Le jeu

L'idée c'est de tirer 2 pages Wikipédia au sort, et d'aller de la 1er à la 2eme , uniquement en utilisant les liens hypertexte présent sur la page, en utilisant le moins de coups possible.

L'idée du jeu n'est pas de moi : il existe même une appli ou un championnat avec des flammes dans le générique !

Exemple :
Je part de la page Python et je doit aller à la page République démocratique du Congo
• Je choisi le lien Pays Bas
• Puis Afrique du Sud
• Puis Afrique australe
• Et République démocratique du Congo

Gagné en 4 coups

Le programme que vous allez écrire sera un « habillage » pour ce jeu

2 Version CLI

Cette étape n’est pas obligatoire, mais recommandée (comme ça vous aurais quand même un livrable, même si votre projet n’aboutit pas)

Dans un premier temps, vous devez écrire un jeu qui fonctionne, mais en « CLI brute »

C’est à dire, en mode texte sans fioritures d’affichage.

Votre affichage doit ressemeler à ça :

```
********************** WikiGame **** tour 1
Départ : Python (langage)
Cible : République démocratique du Congo
Actuellement : Pays-Bas
01 - Pays-Bas (homonymie)
02 - Hollande
03 - néerlandais
04 - Europe de l'Ouest
05 - Belgique
06 - Allemagne
07 - France
08 - île de Saint-Martin
09 - Caraïbes
10 - Monarchie constitutionnelle
11 - 2017
12 - Amsterdam
13 - pouvoirs exécutif
14 - législatif
15 - judiciaire
16 - La Haye
17 - territoires autonomes
18 - Aruba
19 - Curaçao
20 - Saint-Martin
99 - Voir la suite
Votre choix :
```

2.1 Explications

Les 2 bornes (Page de départ et d’arrivé) sont générés grâce à l’URL « random » de Wikipedia :

https://fr.wikipedia.org/wiki/Sp%C3%A9cial:Page_au_hasard

Les choix 01 à 20 sont les 20 premiers liens hypertext de la page (on affiche les liens par page de 20 Car certaines pages en ont plusieurs centaines !)

Attention, ne prendre que les liens de la partie centrale, pas ceux du menu de gauche ni du haut.

Ne pas prendre non plus les liens interne du sommaire

En cas de pagination on ajoute les liens :

```
98 : Page précédente
99 : Page suivante
```

Ces 2 liens ne sont afficher que si besoin (pas de page précédente sur la 1er page, etc ..)

Ils ne changent pas le compteur de tour joués

A chaque tour, vous commencerez l'affichage par effacer l'écran (**cls** / Windows ; **clear** / Linux & Mac)

Attention à tester l'OS sur lequel le programme tourne

3 Version « classe »

Une fois que votre jeu fonctionne, il faut l'embellir !

Vous pouvez soit rester en CLI (Comand Line Interface, ou « mode texte ») et vous améliorez le

rendu avec des librairies comme PyInquirer ou autre

Soit vous passez en GUI (Graphic User Interface) et utilisez des librairies comme TkInter, QT ou l'une de leurs nombreuse alternatives

Soit encore, vous utilisez une librairie comme Eel ou Django qui génèrent des serveurs web et vous permettent d'utiliser le HTML / CSS pour l'affichage

Soit …

En cas de doute, je suis là pour valider votre projet

4 Version « classe + améliorations »

L'idée maintenant est d'enrichir le jeu avec des fonctionnalités.

Quelques idées en vrac (mais vous pouvez / devez ajouter les vôtres)

Attention : Vous ne devez pas toutes les développer : Ce ne sont que des suggestions.

4.1 Mode preview

Le programme affiche, au survol du lien, le résumé Wikipédia de la page

4.2 Mode Historique

En cas de victoire, le programme affiche l'historique des pages utilisées

4.3 Mode HighScore

En cas de victoire, le programme affiche le chemin le plus court possible entre les 2 pages

4.4 Mode timer

Le programme ne compte pas les tours mais le temps.

Et s'arrête si le joueur dépasse n minutes (à paramétrer)

4.5 Mode thématique

Les URLs ne sont pas tirés complètement au sort, mais dans un corpus réduit (Jeux vidéos, Cinéma, ...)

Pour cela, utilisez les portails thématiques Wikipédia.

Ex : Portail du jeu vidéo

- Écrire un second programme qui vas scraper le portail pour en extraire une liste d'URLs qu'il va stocker dans un fichier texte.

- Ajouter un switch -d <fichier> au programme principal.

Si le switch -d est présent, ne pas tirer 2 pages au sort, mais les tirer au sort dans le fichier d'URLs passé en paramétre

Ex :

```
#python dicoMaker.py
Ok, fichier jeuvideo.txt généré

#python wikigame.py -d jeuvideo.txt
********************** WikiGame **** tour 0
Départ : Body Harvest
Cible : Fortnite
Actuellement : Body Harvest
01 - jeu vidéo
02 - DMA Design
03 - Rockstar North
…
```

4.6 Mode duel

Un switch « -n » permet, non pas le lancer le jeu, mais de générer « une graine »

ex :

```
#python wikigame.py -n
lancer "python wikigame.py -s Andasta_benoiti#Henri_Durville"
```

Et si on lance le jeu avec le switch -s le jeu ne tire pas des pages au sort, mais lit les 2 URLs Wikipédia contenu dans le second paramètre (séparées par un #) et les utilise comme début & fin.

Fonctionnement :

Un joueur génère une graine (-n)

Il partage l'URL avec les autres (Discord, ...)

Tout les joueurs lancent le programme avec la graine (-s) en même temps et le 1er arrivé à gagné.

Amélioration : La graine est encodé en base64 pour ne pas permettre aux joueurs de « voir » les URLs qu'elle contient

```
#python wikigame.py -n
lancer "python wikigame.py -s QW5kYXN0YV9iZW5vaXRpI0hlbnJpX0R1cnZpbGxl"
```

4.7 <insert here idée géniale>

A vous d'imaginer vos propres extensions

5 Livrable

Un dépôt Git (GitLab (bien), GitHub (bouh) …) avec un README.md qui explique comment installer le jeu (dépendances PIP, …)

Pour ceux qui veulent garder leur dépôt privé : Ok, mais pensez à m'inviter avec des droits suffisants pour que je voie le code.

Mon identifiant GitLab : https://gitlab.com/arrobe

Mon identifiant GitHub : https://github.com/ArrobeHugues

Vous n'avez pas à livrer 3 versions du jeu : Seule la plus aboutie est attendue

Date de rendu : Avant le dimanche 30 juin, minuit

Rendu par MP Discord ou par mail (hugues.levasseur@arrobe.fr)

6 Barème

6.1 Qualité du code (8 points)

Concerne :

- La qualité d'écriture du code (optimisé, POO, écrit / importé, …)
  - Précision sur POO : Pas obligé, mais apprécié si bien fait
  - Précision sur écrit / importé : On est en 2024, vous n'allez pas réécrire vous même un serveur web ou une fonction qui calcule une racine carrée ...
- La maintenabilité du code (Norme de nommage des variables / fonctions ; commentaires, ...)
- La résistance du jeu aux bugs, erreurs de saisie, …

6.2 Fonctionnalités (8 points)

Concerne :

- Réalisation compléte ou partielle du TP
- Richesse fonctionnelle (Fonctionnalités ajoutées)
- Qualité de la réalisation (nombre d'options ajoutées)
- Originalité des options ajoutées

6.3 Ergonomie (4 points)

Concerne :

- Facilité d'installation & clarté du README.MD
- Fluidité & facilité d'utilisation

- Ergonomie
- Rendu graphique (si GUI)