

## Rapport du projet Python : ESGI Animes



Sayah MEDELLEL  
Moussa CAMARA  
Oussama ABDELHEDI

## Le rapport final

## **L'équipe du projet :**

Sayah MEDELLEL, Moussa CAMARA, Oussama ABDELHEDI

## **Période :**

Semestre 2 (2021)

## **Établissement :**

ESGI, École Supérieure de Génie Informatique

## **Sujet :**

Scraping du site Gum-Gum Streaming et création d'un site Django avec les données collectées.

## **Professeur en charge :**

Mr. Benjamin RAYNAL

## Sommaire

Ce rapport est composé :

- D'une partie présentation générale du projet, avec une introduction, des points non résolus, des difficultés rencontrées et d'un bilan global
- D'une annexe technique composée des dossiers concepteur et utilisateur
- D'une annexe informelle composée d'une bibliographie et d'un glossaire

Le dossier concepteur présente la conception du projet ainsi qu'une analyse de l'application regroupant plusieurs éléments :

- Une description de la structure du projet
- Une description des méthodes principales
- Une description des choix d'implémentation et d'autres détails techniques

Le dossier utilisateur regroupe l'ensemble des informations permettant de comprendre l'installation et l'utilisation de l'application.

La bibliographie regroupe les références des documents qui nous ont aidés pour réaliser le projet.

## Introduction

Dans le cadre de la réalisation d'un projet en Python, nous avons réalisé une application qui permet de récupérer les données du site Gum-Gum-Streaming et de les analyser. Gum-Gum Streaming est un site contenant des épisodes d'animés. Les données que nous allons collecter sont : les fiches descriptives des animés et la liste des épisodes pour chaque animé. Une fois les données collectées nous allons les mettre sur un site développé en Django.

Le projet est scindé en deux parties :

- Data Acquisition : automatisation d'un processus permettant de recueillir les données de plusieurs séries et de les regrouper dans plusieurs fichiers structurés.
- Data Visualisation : représentation graphique des données afin de communiquer les informations de manière claire et efficace

Pour réaliser ce projet, l'équipe est composée de trois membres :

- Sayah MEDELLEL
- Moussa CAMARA
- Oussama ABDELHEDI

## Difficultés rencontrées

Nous avons rencontré quelques difficultés (humaines et techniques) pour la réalisation du projet.

Tout d'abord, on peut évoquer le manque de communication, parfois nous ne recevions pas les messages en temps et en heure, ce qui est un inconvénient lorsqu'il faut modifier le travail. L'absence physique et les échanges à distance ont favorisé les pertes d'informations et ont pu créer une sorte de manque de cohésion. Les idées étaient plus difficiles à exprimer et il s'ensuivit des incompréhensions entre les membres de l'équipe.

De plus, nous avions des projets à rendre en parallèle donc nous avons eu une contrainte de temps.

Enfin, nous avons fait face à plusieurs problèmes techniques (notamment au niveau de la conception). Concernant les installations, nous avons également eu quelques problèmes à installer et configurer correctement certains outils. Il était nécessaire de comprendre tous les éléments qui permettent l'installation et la configuration de ces outils, dont le fait de trouver les dernières libs à jour (précompilés).

Afin de surmonter toutes ces difficultés nous avons dû nous entraider, communiquer plus régulièrement entre nous à travers les systèmes de messageries et nous aider des différents documents qui étaient à notre disposition (cours, sites web).

## Bilan

La réalisation de ce projet a permis au groupe d'acquérir des compétences sur la programmation en python et notamment dans l'extraction de données mais également sur la capacité à donner un sens aux données (visualisation sur un site web avec filtrage des données). Ce projet nous a également permis de développer nos compétences sur plusieurs outils et technologies qui étaient encore inconnu pour nous.

## Annexe technique

## I) Dossier concepteur

### **Introduction**

Ce document explique la phase d'avant-projet, phase qui va permettre de définir les objectifs clairs de l'application et notamment la manière dont celle-ci va être réalisée. Cette phase est nécessaire afin de comprendre au mieux le sujet posé et de l'aborder convenablement. Il est clair que cela va permettre de simplifier les étapes de réalisation par la suite. Globalement, elle va permettre de structurer, organiser et planifier le projet.

### **Spécification détaillée de la structure du système**

Afin de comprendre la méthode de conception appliquée, nous avons effectué une description détaillée du système, réparti en un aspect technique et un aspect complémentaire.

#### **A) Aspect technique**

L'application est composée d'un ensemble de fonctions qui permettent le scraping de Gum-Gum Streaming. Globalement, les fonctions vont permettre à un utilisateur :

- De récupérer toutes les données des animés du site Gum-Gum Streaming
- Les insérer dans une base de données
- Développer un site web avec le framework Django et importer les données de la base dessus.

#### **B) Aspect complémentaire**

Globalement, le but d'une telle structure est de fournir un programme qui répond à plusieurs critères essentiels : flexibilité, durabilité et adaptabilité. D'ailleurs le programme conçu permet d'assurer complètement l'aspect de la dynamique. En outre, le fonctionnement du programme et les résultats s'adaptent par rapport aux présentes dans le site. La date n'influe pas sur le comportement du programme, ce qui encore une fois est intéressant pour l'utilisateur.



## **C) Outils et technologies**

### **1) Techniques**

Le langage utilisé pour réaliser le programme de récupération des données du site Gum-Gum Streaming est le Python. Des requêtes SQL ont également été créées afin d'établir des tests sur les données présentes dans la base de données SQLite.

### **2) Ressources immatérielles**

Plusieurs ressources ont été utilisées, notamment des logiciels et applications permettant :

- Le développement du scraper et du site (PyCharm, Visual Studio Code)
- Le stockage des données (SQLite)
- Le déploiement du site web (Heroku)

## **D) Comportement**

Nous avons vu précédemment que le programme a été conçu de manière à assurer l'aspect de la dynamique. Pour assurer un tel aspect, le programme a été conçu avec une logique d'automatisation, limitant ainsi les interactions entre l'utilisateur et le programme. Nous allons détailler le fonctionnement du programme et des fonctions. Pour rappel, le projet est scindé en trois parties :

- Data Acquisition
- Data Visualisation

Chaque partie présente une méthodologie et un mode de fonctionnement différent.

### **1) Data Acquisition**

L'objectif dans cette partie est de récupérer les données concernant les fiches descriptives des animés et la liste des épisodes pour chaque animé du site.

Figure 1 : Présentation de la fiche de l'animé Shingeki no Kyojin



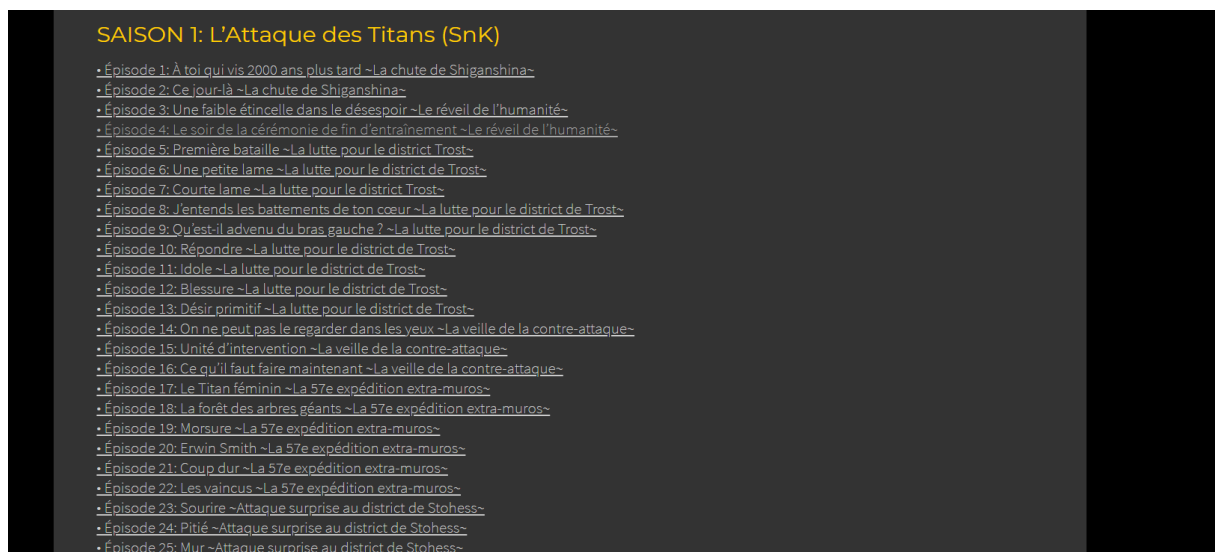
ACCUEIL VF VOSTFR FILMS AUTRES BOUTIQUES SHOP Exemple : Code Geass VOSTFR

## Shingeki No Kyojin (Attaque des Titans) VF

Auteur: Isayama Hajime  
Type: Shonen  
Genre: Action, drame, fantastique, horreur, psychologique, surnaturel, mystère  
Studio d'animation: Wit Studio  
Année de production: 2018  
Durée: 59 épisodes  
Statut: En pause (saison 4 annoncée)  
Note: ★★★★★

**Synopsis:** Il y a 107 ans, les Titans ont presque exterminé la race humaine. Ces Titans mesurent principalement une dizaine de mètres et ils se nourrissent d'humains. Les humains ayant survécus à cette extermination ont construit une cité fortifiée avec des murs d'enceinte de 50 mètres de haut pour pouvoir se protéger des Titans. Pendant 100 ans les humains ont connu la paix. Eren est un jeune garçon qui rêve de sortir de la ville pour explorer le monde extérieur. Il mène une vie paisible avec ses parents et sa sœur Mikasa dans le district de Shiganshina. Mais un jour de l'année 845, un Titan de plus de 60 mètres de haut apparaît. Il démolit une partie du mur du district de Shiganshina et provoque une invasion de Titans. Eren verra sa mère se faire dévorer sous ses yeux sans rien pouvoir faire. Il décidera après ces événements traumatisants de s'engager dans les forces militaires de la ville avec pour but d'exterminer tous les Titans qui existent.

Figure 2 : Présentation de la liste des épisodes de l'animé Shingeki no Kyojin



### SAISON 1: L'Attaque des Titans (SnK)

- Episode 1: À toi qui vis 2000 ans plus tard ~La chute de Shiganshina~
- Episode 2: Ce jour-là ~La chute de Shiganshina~
- Episode 3: Une faible étincelle dans le désespoir ~Le réveil de l'humanité~
- Episode 4: Le soir de la cérémonie de fin d'entraînement ~Le réveil de l'humanité~
- Episode 5: Première bataille ~La lutte pour le district Trost~
- Episode 6: Une petite lame ~La lutte pour le district de Trost~
- Episode 7: Courte lame ~La lutte pour le district Trost~
- Episode 8: J'entends les battements de ton cœur ~La lutte pour le district de Trost~
- Episode 9: Qu'est-il advenu du bras gauche? ~La lutte pour le district de Trost~
- Episode 10: Répondre ~La lutte pour le district de Trost~
- Episode 11: Idole ~La lutte pour le district de Trost~
- Episode 12: Blessure ~La lutte pour le district de Trost~
- Episode 13: Désir primitif ~La lutte pour le district de Trost~
- Episode 14: On ne peut pas le regarder dans les yeux ~La veille de la contre-attaque~
- Episode 15: Unité d'intervention ~La veille de la contre-attaque~
- Episode 16: Ce qu'il faut faire maintenant ~La veille de la contre-attaque~
- Episode 17: Le Titan féminin ~La 57e expédition extra-muros~
- Episode 18: La forêt des arbres géants ~La 57e expédition extra-muros~
- Episode 19: Morsure ~La 57e expédition extra-muros~
- Episode 20: Erwin Smith ~La 57e expédition extra-muros~
- Episode 21: Coup dur ~La 57e expédition extra-muros~
- Episode 22: Les vaincus ~La 57e expédition extra-muros~
- Episode 23: Sourire ~Attaque surprise au district de Stohess~
- Episode 24: Pitié ~Attaque surprise au district de Stohess~
- Episode 25: Mur ~Attaque surprise au district de Stohess~

L'objectif est donc de récupérer certaines données des deux fiches.

Les données de la fiche descriptive seront contenues dans un dictionnaire qui sera contenu dans un fichier json.

Les données de la liste d'épisodes seront contenues dans une liste de dictionnaires qui sera contenu dans un fichier json.

Figure 3 : Tableau représentant la liste des données à récupérer selon les fiches

Fiche	Données à récupérer
Animé	Id, Url, Nom, Langue, Note moyenne, Auteurs, Type, Genre, Studio d'animation, Année de production, Nombre d'épisodes, Statut, Synopsis
Episode	Url de l'épisode, Url de l'animé, Nom de l'épisode

Dans ce dossier nous avons les scripts :

**collect\_urls.py** : Récupère la liste des animés à scraper pour les pages :

<https://gum-gum-streaming.com/vf/> | <https://gum-gum-streaming.com/vostfr/>

-> Génère un fichier json pour chaque page dans le dossier anime\_list\_collected.

**aggregate\_urls.py** : Agrège tous les fichiers obtenus avec le script collect\_urls.py en un fichier json.

-> Génère un fichier dans le dossier anime\_list\_aggregated.

**filter\_urls.py** : Filtre les urls afin d'obtenir un fichier json avec des dictionnaires contenant les urls de tout les animés que nous collecter ainsi qu'une clé « collected » initialisée à « no » qui permet de savoir si une url d'animé a été scrapée ou non.

-> Génère un fichier json dans le dossier anime\_urls\_to\_collect.

**data\_collector.py** : Contient l'ensemble des méthodes permettant de collecter les infos d'une fiche descriptive et la liste des épisodes d'un animé

**collect\_anime.py** : Script de test qui collecte les infos d'une fiche descriptive et la liste des épisodes d'un animé seulement. (Permet donc de tester les méthodes de data\_collector.py).

-> Génère un fichier json dans le dossier anime\_data et un fichier json dans le dossier anime\_episodes\_data.

**collect\_animes.py** : Collecte les infos de la fiche descriptive et la liste des épisodes des animés contenues dans la liste obtenue avec le script `filter_urls.py`.

-> Génère plusieurs fichiers json dans le dossier *animes\_data* et plusieurs fichiers json dans le dossier *animes\_episodes\_data*.

**aggregate\_files.py** : Agrège le contenu des dossiers *animes\_data* et *animes\_episodes\_data*.

-> Génère un fichier dans le dossier *merged\_animes\_data* et un fichier dans le dossier *merged\_animes\_episodes\_data*.

## 2) Data Visualisation

L'objectif de cette partie est de modéliser les données disponibles dans la base de données. Globalement, les données stockées représentent les informations relatives aux fiches descriptives des animés et aux listes des épisodes.

Nous avons décidé de faire un site en Django sur lequel nous avons pu accéder aux animés via différentes rubriques (Animés en cours ou terminé, en VF ou en VOSTFR) et une barre de recherche.

Pour ce faire nous avons utilisé :

- SQLite afin de créer la base de données et de faire des requêtes SQL dessus.
- Le Framework Django afin de développer le site web.

Figure 4 : Page du site web contenant les animés ayant le statut « En cours »

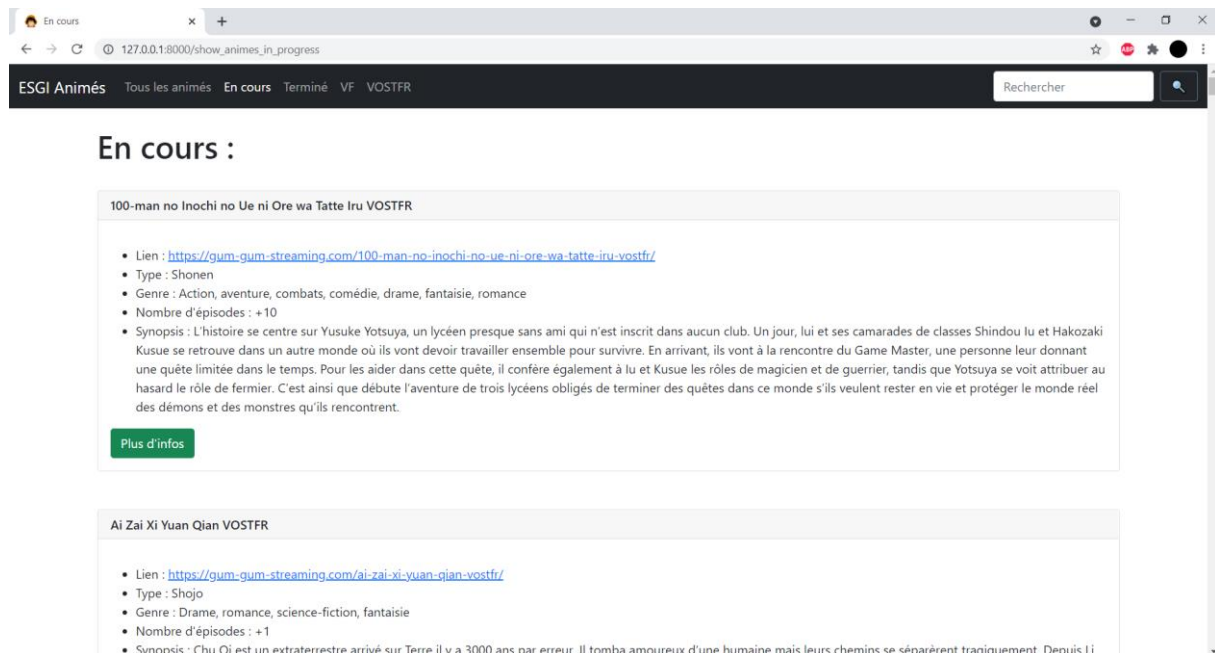


Figure 5 : Page contenant toutes les informations d'un animé lorsqu'on clique sur « Plus d'infos »

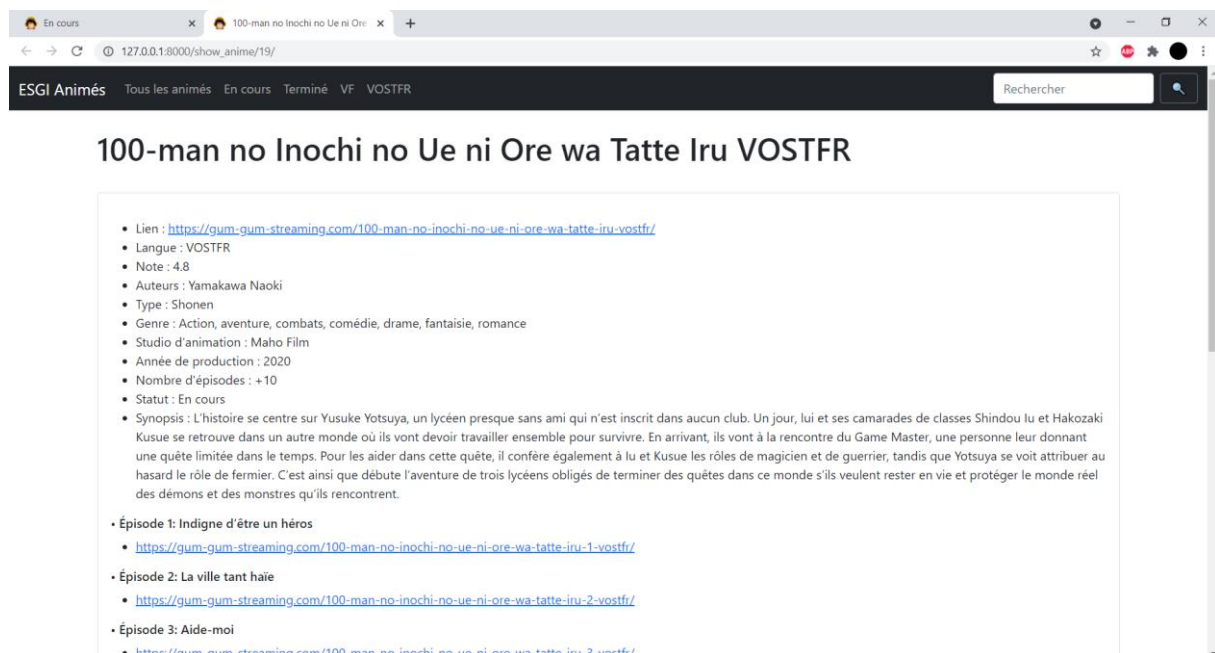
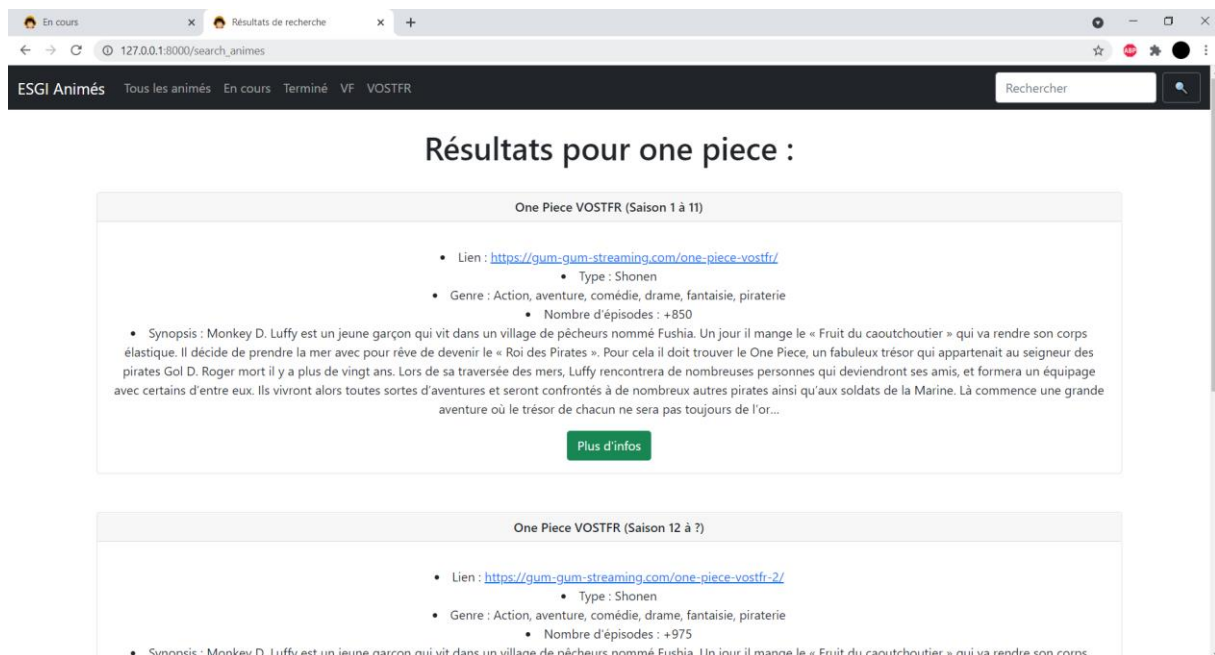


Figure 6 : Page contenant les résultats d'une recherche

## II) Dossier utilisateur

### Introduction

Le dossier utilisateur permet à l'utilisateur de lui apporter tous les renseignements nécessaires et suffisants pour une bonne utilisation et compréhension de l'application. Il explique à l'utilisateur comment utiliser le programme, corriger les possibles erreurs et comprendre les résultats du programme.

### Présentation de l'application

#### A) Fonctionnalités

Le produit final est une application qui permet de scraper le site Gum-Gum Streaming, formater les données et enfin les afficher sous une forme avec plus choix de filtrage via le site en Django.

#### B) Mode d'utilisation

Les données concernant les séries et les avis sont récupérés grâce à l'exécution des scripts du dossier data\_acquisition dans un ordre précis. Durant l'exécution les fichiers avec les données des fiches descriptives et des listes d'épisodes sont créés. Un autre script permet « l'assemblage » de ces fichiers, formant ainsi deux fichiers finaux : un pour l'ensemble des fiches descriptives et un autre pour l'ensemble des listes d'épisodes.

Ainsi, pour utiliser le scraper il faut lancer les scripts dans l'ordre suivant :

- collect\_urls.py
- aggregate\_urls.py
- filter\_urls.py
- collect\_animes.py
- aggregate\_files.py

Pour tester le scraper il suffit de lancer le script :

- collect\_anime.py

Concernant la partie visualisation il faut :

- Installer les librairies nécessaires à l'exécution du site avec la commande :  
`pip install -r requirements.txt`
- Lancer la commande `python manage.py runserver` à partir du dossier *data\_vizualisation*.
- Accéder au site via l'url <http://127.0.0.1:8000/>



## **Annexe informelle**

## Bibliographie

- Cours de Mr. Raynal
- <https://stackoverflow.com>
- Documentation sur Django : <https://docs.djangoproject.com/en/3.2/>
- Playlist sur le développement de site web en Django :  
[https://www.youtube.com/watch?v=LxEFgfPdhDg&list=PLEn9o0UAh\\_cCszR1kXFAe-4lxkJjCnAiH](https://www.youtube.com/watch?v=LxEFgfPdhDg&list=PLEn9o0UAh_cCszR1kXFAe-4lxkJjCnAiH)