



Актуальные зарплаты в IT

110k

100k

150k

120k

180k



empenoso

25 ноя 2024 в 03:23

Как за 6 промтов к ChatGPT создать Python скрипт, скачивающий видео с YouTube для просмотра на телевизоре через Kodi

Простой

18 мин

20K

Open source*, Умный дом, Искусственный интеллект, Python*

Кейс

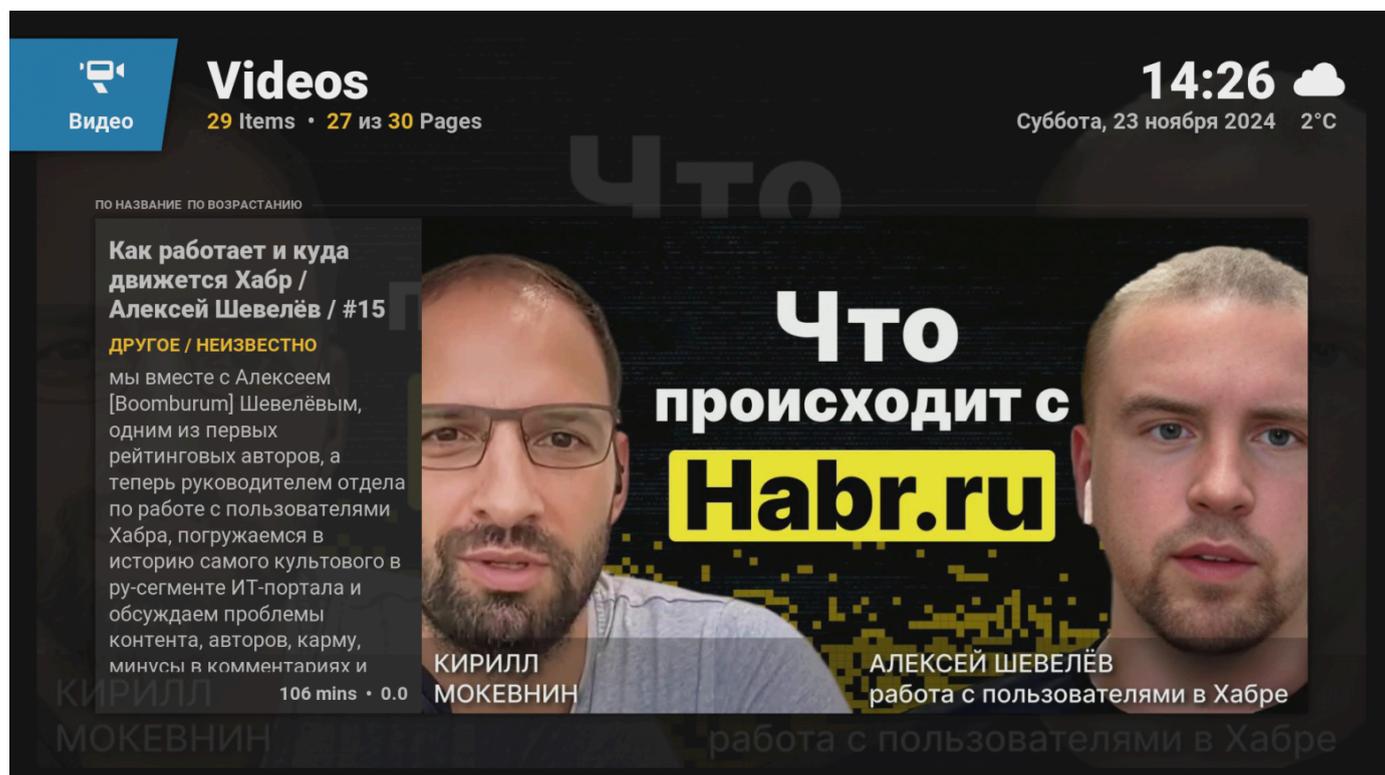
Последние месяцы Ютуб работает с перебоями: через сеть мобильного оператора вроде работает, но при подключении к сети проводного интернет-провайдера чаще всего ничего посмотреть нельзя, хотя в новостях что-то промелькнуло насчёт возобновления работы, но лично у меня ничего не поменялось.

Я [использую свободный медицентр Kodi](#) для просмотра своей коллекции фильмов и сериалов, создавая домашний кинотеатр без ежемесячной абонентской платы. Это достигается за счёт того, что все файлы уже хранятся на собственном сетевом хранилище с локальным доступом или записаны прямо в памяти Kodi, который может быть установлен практически на любой платформе.

Раньше на Kodi можно было напрямую отправить YouTube ссылку и посмотреть любое видео. С августа 2024 года этот вариант больше не работает. Короткие видео можно посмотреть с телефона, но вот длинные интервью или подкасты так смотреть или слушать неудобно. К тому же телевизором можно управлять через Алису - она подключена к Home Assistant.

Конечно, чтобы скачать видео с Ютуба существует множество программ, но скачивается только сам видео файл и он будет отображаться в базе Kodi без обложки и описания. Вручную создавать обложку и описание и тратить на это своё время не вариант - зачем?

Разбираюсь как за 6 промтов к ChatGPT создать Python скрипт, скачивающий видео с YouTube и генерирующий описание и фанарт для Kodi.



Итог работы скрипта

Статья получилась длинная, потому что полностью приведены запросы и ответы, поэтому сделал содержание. После номера промта идёт результат выполнения текущего ответа от ChatGPT.

Промт №1: предлагаемая библиотека youtube не работает

Промт №2: предлагаемая библиотека yt-dlp требует аутентификацию для скачивания

Промт №3: ошибки в коде

Промт №4: видео и звуковая дорожка по отдельности

Промт №5: Kodi не распознаёт описания в .nfo файле

Промт №6: заработало

Итоговый вариант на [GitHub](#)

Промт №1: предлагаемая библиотека youtube не работает

Действуй как опытный программист Python. Напиши скрипт, который будет скачивать видео с сервиса Ютуб в максимальном качестве.

На входе ссылка например "https://www.youtube.com/watch?v=_lvcEUi1MtE".

На выходе .mkv файл.

Дополнительно сгенерируй .nfo файл описания этого видео используя данные заголовка, описания и обложки видео. .nfo файл нужен чтобы загрузить эти данные в базу Kodi.

Добавляй все комментарии на русском языке, продолжайте со мной общение на английском.

Вот скрипт Python, использующий библиотеку youtube для загрузки видео YouTube в максимальном качестве и создания файла .nfo с метаданными для Kodi. Для работы с видео YouTube вам нужно будет установить youtube:

```
pip install youtube
```

По запросу в сценарий включены комментарии на русском языке:

```
import os
from youtube import YouTube
import requests

# Функция для загрузки видео с YouTube
def download_video(url, output_dir):
    # Создаем объект YouTube для указанного URL
    yt = YouTube(url)

    # Получаем поток видео с максимальным разрешением
    video_stream = yt.streams.get_highest_resolution()

    # Устанавливаем имя выходного файла
```

```
video_title = yt.title
output_path = os.path.join(output_dir, f"{video_title}.mkv")

# Загружаем видео
print(f"Загружается видео: {video_title}")
video_stream.download(output_path=output_path)
print(f"Видео успешно загружено: {output_path}")

return output_path, yt

# Функция для создания .nfo файла
def create_nfo_file(video_path, yt, output_dir):
    # Получаем метаданные: заголовок, описание, обложка
    title = yt.title
    description = yt.description
    thumbnail_url = yt.thumbnail_url

    # Загружаем изображение обложки
    thumbnail_path = os.path.join(output_dir, "cover.jpg")
    response = requests.get(thumbnail_url)
    with open(thumbnail_path, "wb") as f:
        f.write(response.content)

    # Создаем содержимое .nfo файла
    nfo_content = f"""<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<movie>
  <title>{title}</title>
  <plot>{description}</plot>
  <thumb>{thumbnail_path}</thumb>
</movie>
"""

    nfo_path = os.path.splitext(video_path)[0] + ".nfo"

    # Записываем содержимое в .nfo файл
    with open(nfo_path, "w", encoding="utf-8") as nfo_file:
        nfo_file.write(nfo_content)

    print(f".nfo файл успешно создан: {nfo_path}")
    return nfo_path

# Основная программа
def main():
    # URL видео YouTube
```

```
url = "https://www.youtube.com/watch?v=HbsPPpeACvI"

# Каталог для сохранения файлов
output_dir = "./downloads"
os.makedirs(output_dir, exist_ok=True)

try:
    # Загружаем видео
    video_path, yt = download_video(url, output_dir)

    # Создаем .nfo файл
    create_nfo_file(video_path, yt, output_dir)

except Exception as e:
    print(f"Произошла ошибка: {e}")

if __name__ == "__main__":
    main()
```

Объяснение кода

1. Загрузка видео:

- Использует `pytube` для загрузки и скачивания видео в самом высоком разрешении.
- Выходной файл сохраняется в формате `.mkv`.

2. Создание файла `.nfo`:

- Извлекает метаданные, такие как заголовок, описание и URL миниатюры.
- Загружает миниатюру видео как `cover.jpg` и включает ее в файл `.nfo`.
- Сохраняет файл `.nfo` в том же каталоге, что и видео.

3. Удобное управление каталогами:

- Обеспечивает организованную структуру каталогов с папкой `downloads`.

4. Русские комментарии:

- Все комментарии написаны на русском языке, чтобы подробно объяснить каждый шаг.

Дайте мне знать, если вам нужны улучшения или объяснения!

Промт №2: предлагаемая библиотека yt-dlp требует аутентификацию для скачивания

Эта библиотека не работает и возвращает ошибку:

```
$ pytube https://youtube.com/watch?v=2lAe1cqCOXo

+ CategoryInfo          : ObjectNotFound: (String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException
D:\SynologyDrive\docs\2024_11_YouTube downloader> pytube https://youtube.com/watch?v=2l
Loading video...
Traceback (most recent call last):
  File "<frozen runpy>", line 198, in _run_module_as_main
  File "<frozen runpy>", line 88, in _run_code
  File "C:\Users\михаил\AppData\Local\Programs\Python\Python312\Scripts\pytube.exe\__ma
  File "C:\Users\михаил\AppData\Local\Programs\Python\Python312\Lib\site-packages\pytub
    _perform_args_on_youtube(youtube, args)
  File "C:\Users\михаил\AppData\Local\Programs\Python\Python312\Lib\site-packages\pytub
    download_highest_resolution_progressive(
  File "C:\Users\михаил\AppData\Local\Programs\Python\Python312\Lib\site-packages\pytub
    stream = youtube.streams.get_highest_resolution()
               ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\михаил\AppData\Local\Programs\Python\Python312\Lib\site-packages\pytub
    return StreamQuery(self.fmt_streams)
               ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\михаил\AppData\Local\Programs\Python\Python312\Lib\site-packages\pytub
    stream_manifest = extract.apply_descrambler(self.streaming_data)
               ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\михаил\AppData\Local\Programs\Python\Python312\Lib\site-packages\pytub
    self.bypass_age_gate()
  File "C:\Users\михаил\AppData\Local\Programs\Python\Python312\Lib\site-packages\pytub
    innertube_response = innertube.player(self.video_id)
               ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

```

File "C:\Users\михаил\AppData\Local\Programs\Python\Python312\Lib\site-packages\pytube
    return self._call_api(endpoint, query, self.base_data)
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
File "C:\Users\михаил\AppData\Local\Programs\Python\Python312\Lib\site-packages\pytube
    response = request._execute_request(
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
File "C:\Users\михаил\AppData\Local\Programs\Python\Python312\Lib\site-packages\pytube
    return urlopen(request, timeout=timeout) # nsec
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
File "C:\Users\михаил\AppData\Local\Programs\Python\Python312\Lib\urllib\request.py",
    return opener.open(url, data, timeout)
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
File "C:\Users\михаил\AppData\Local\Programs\Python\Python312\Lib\urllib\request.py",
    response = meth(req, response)
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
File "C:\Users\михаил\AppData\Local\Programs\Python\Python312\Lib\urllib\request.py",
    response = self.parent.error(
    ^^^^^^^^^^^^^^^^^^^^^^^^^
File "C:\Users\михаил\AppData\Local\Programs\Python\Python312\Lib\urllib\request.py",
    return self._call_chain(*args)
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
File "C:\Users\михаил\AppData\Local\Programs\Python\Python312\Lib\urllib\request.py",
    result = func(*args)
    ^^^^^^^^^^^^^
File "C:\Users\михаил\AppData\Local\Programs\Python\Python312\Lib\urllib\request.py",
    raise HTTPError(req.full_url, code, msg, hdrs, fp)
urllib.error.HTTPError: HTTP Error 403: Forbidden

```

Ошибка, с которой вы столкнулись, указывает на то, что библиотека `pytube` вышла из строя из-за изменения API YouTube или структуры потоковой передачи, что является распространенной проблемой для библиотек, полагающихся на обратную разработку внутренних механизмов YouTube. Вот почему такие библиотеки часто ломаются, пока их сопровождающие не обновят их.

Альтернативное, надежное решение — использовать `yt-dlp`, мощный и активно поддерживаемый форк `youtube-dl`. Он хорошо подходит для загрузки видео с YouTube и других платформ. Вот как это реализовать:

Шаги:

1. Установите `yt-dlp` :

```
pip install yt-dlp
```

2. Скрипт Python:

Ниже приведен обновленный скрипт, использующий `yt-dlp` для загрузки видео в максимальном качестве и создания файла `.nfo` :

```
import os
import subprocess
import json

# Функция для загрузки видео
def download_video(url, output_dir):
    # Путь для выходного видеофайла
    output_template = os.path.join(output_dir, "%(title)s.%(ext)s")

    # Команда yt-dlp для загрузки видео в максимальном качестве
    command = [
        "yt-dlp",
        "-f", "bestvideo+bestaudio/best", # Выбор лучшего видео и аудио
        "--merge-output-format", "mkv", # Сохранение в MKV
        "-o", output_template, # Шаблон имени файла
        "--write-info-json", # Сохранение метаданных
        url
    ]

    print("Загружается видео...")
    subprocess.run(command, check=True)
    print("Видео успешно загружено.")

    # Находим JSON файл с метаданными
    info_json_path = next(
        (os.path.join(output_dir, f) for f in os.listdir(output_dir) if f.endswith(".info.json"))
    )
    if not info_json_path:
        raise FileNotFoundError("Файл метаданных .info.json не найден.")

    return info_json_path

# Функция для создания .nfo файла
```

```
def create_nfo_file(info_json_path):
    # Читаем данные из JSON файла
    with open(info_json_path, "r", encoding="utf-8") as f:
        metadata = json.load(f)

    title = metadata.get("title", "Без названия")
    description = metadata.get("description", "Нет описания")
    thumbnail = metadata.get("thumbnail", "")

    # Загружаем обложку, если доступна
    cover_path = None
    if thumbnail:
        cover_path = os.path.splitext(info_json_path)[0] + ".jpg"
        with open(cover_path, "wb") as f:
            f.write(requests.get(thumbnail).content)

    # Создаем содержимое .nfo файла
    nfo_content = f"""<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<movie>
  <title>{title}</title>
  <plot>{description}</plot>
  <thumb>{cover_path or "Нет обложки"}</thumb>
</movie>
"""

    # Сохраняем .nfo файл
    nfo_path = os.path.splitext(info_json_path)[0] + ".nfo"
    with open(nfo_path, "w", encoding="utf-8") as nfo_file:
        nfo_file.write(nfo_content)

    print(f".nfo файл успешно создан: {nfo_path}")
    return nfo_path

# Основная программа
def main():
    # URL видео YouTube
    url = "https://www.youtube.com/watch?v=HbsPPpeACvI"

    # Каталог для сохранения файлов
    output_dir = "./downloads"
    os.makedirs(output_dir, exist_ok=True)

    try:
```

```
# Загружаем видео и метаданные
info_json_path = download_video(url, output_dir)

# Создаем .nfo файл
create_nfo_file(info_json_path)

except Exception as e:
    print(f"Произошла ошибка: {e}")

if __name__ == "__main__":
    main()
```

Основные характеристики:

1. yt-dlp :

- Автоматически извлекает видео и аудио наилучшего качества и объединяет их в файл `.mkv` .
- Создает файл `.info.json` с метаданными (например, заголовок, описание, миниатюра).

2. `.nfo` Файл:

- Анализирует файл `.info.json` на предмет метаданных видео.
- Загружает миниатюру и включает ее путь в файл `.nfo` .

3. Обработка ошибок:

- Обеспечивает корректную остановку скрипта в случае возникновения проблем (например, отсутствие метаданных JSON).

4. Повторно используемая структура:

- Скрипт можно легко адаптировать для пакетной загрузки или расширить для дополнительных функций.

Почему yt-dlp?

- Активно поддерживается и регулярно обновляется.
- Поддерживает широкий спектр форматов и опций.
- Надежен для загрузки видео YouTube, даже когда другие библиотеки не справляются.

Запустите скрипт и дайте мне знать, как он работает!

Промт №3: ошибки в коде

Какая-то ошибка - что мне сделать?

```
[Running] python -u "d:\SynologyDrive\docs\2024_11_YouTube downloader\YouTube_download.  
Загружается видео...  
[youtube] Extracting URL: https://www.youtube.com/watch?v=HbsPPpeACvI  
[youtube] HbsPPpeACvI: Downloading webpage  
[youtube] HbsPPpeACvI: Downloading ios player API JSON  
[youtube] HbsPPpeACvI: Downloading mweb player API JSON  
ERROR: [youtube] HbsPPpeACvI: Sign in to confirm you're not a bot. Use --cookies-from-  
Произошла ошибка: Command '['yt-dlp', '-f', 'bestvideo+bestaudio/best', '--merge-output  
[Done] exited with code=0 in 9.306 seconds
```

Ошибка указывает на то, что YouTube требует дополнительной проверки (например, входа в систему) для загрузки видео. Обычно это происходит, когда:

1. Видео имеет возрастные ограничения или помечено как требующее дополнительной проверки.
2. YouTube подозревает необычную активность и запускает проверку CAPTCHA или бота.

Чтобы решить эту проблему, вам необходимо предоставить аутентифицированные файлы cookie для `yt-dlp`, что обходит эти ограничения.

Решение: Используйте файлы cookie браузера

Шаг 1: Экпортируйте файлы cookie из браузера

1. Установите расширение **Cookies.txt** для своего браузера:

- [Chrome](#)
- [Firefox](#)

2. Перейдите на YouTube и **войдите в свою учетную запись**.

3. Откройте видео, которое хотите загрузить, и используйте расширение для **экспорта файлов cookie**. Сохраните файл cookie (например, `cookies.txt`) в рабочем каталоге.

Шаг 2: Измените скрипт

Обновите скрипт, чтобы передать файл cookie в `yt-dlp` :

```
def download_video(url, output_dir, cookies_file):
    # Путь для выходного видеофайла
    output_template = os.path.join(output_dir, "%(title)s.%(ext)s")

    # Команда yt-dlp для загрузки видео с использованием cookies
    command = [
        "yt-dlp",
        "-f", "bestvideo+bestaudio/best", # Выбор лучшего видео и аудио
        "--merge-output-format", "mkv", # Сохранение в MKV
        "-o", output_template, # Шаблон имени файла
        "--write-info-json", # Сохранение метаданных
        "--cookies", cookies_file, # Передача cookies для аутентификации
        url
    ]

    print("Загружается видео...")
    subprocess.run(command, check=True)
    print("Видео успешно загружено.")
```

Вызовите функцию, указав путь к файлу cookie:

```
# Основная программа
def main():
    url = "https://www.youtube.com/watch?v=HbsPPpeACvI"
    output_dir = "./downloads"
    cookies_file = "./cookies.txt" # Укажите путь к файлу cookies

    os.makedirs(output_dir, exist_ok=True)

    try:
        info_json_path = download_video(url, output_dir, cookies_file)
        create_nfo_file(info_json_path)
    except Exception as e:
        print(f"Произошла ошибка: {e}")
```

Шаг 3: Запуск скрипта

1. Сохраните файл `cookies.txt` в том же каталоге, что и ваш скрипт, или укажите правильный путь.
2. Запустите скрипт еще раз. Теперь он должен работать без дополнительной проверки.

Примечания

1. **Сохраняйте файлы cookie в безопасности:** ваш файл cookie содержит конфиденциальные данные. Не делитесь им и не оставляйте его доступным для других.
2. **Обновление файлов cookie:** если срок действия файлов cookie истек, повторите шаг 1, чтобы создать новый `cookies.txt`.
3. **Альтернатива:** вы можете использовать параметр `--cookies-from-browser` для прямого получения файлов cookie из вашего браузера, если он поддерживается.

Сообщите мне, если это решит вашу проблему!

Промт №4: видео и звуковая дорожка по отдельности

Приведи пожалуйста полный файл ответа. Нет ошибки, но и нет загруженного видео:

```
import sys
import time
sys.stdout.reconfigure(encoding='utf-8')

import os
import subprocess
import json

def download_video(url, output_dir, cookies_file):
    # Путь для выходного видеофайла
    output_template = os.path.join(output_dir, "%(title)s.%(ext)s")

    # Команда yt-dlp для загрузки видео с использованием cookies
    command = [
        "yt-dlp",
        "-f", "bestvideo+bestaudio/best", # Выбор лучшего видео и аудио
        "--merge-output-format", "mkv", # Сохранение в MKV
        "-o", output_template, # Шаблон имени файла
        "--write-info-json", # Сохранение метаданных
        "--cookies", cookies_file, # Передача cookies для аутентификации
        url
    ]

    print("Загружается видео...")
    subprocess.run(command, check=True)
    print("Видео успешно загружено.")

    # Находим JSON файл с метаданными
    info_json_path = next(
        (os.path.join(output_dir, f) for f in os.listdir(output_dir) if f.endswith(".info.json"))
    )
    if not info_json_path:
        raise FileNotFoundError("Файл метаданных .info.json не найден.")

    return info_json_path

# Функция для создания .nfo файла
def create_nfo_file(info_json_path):
    # Читаем данные из JSON файла
    with open(info_json_path, "r", encoding="utf-8") as f:
```

```
metadata = json.load(f)

title = metadata.get("title", "Без названия")
description = metadata.get("description", "Нет описания")
thumbnail = metadata.get("thumbnail", "")

# Загружаем обложку, если доступна
cover_path = None
if thumbnail:
    cover_path = os.path.splitext(info_json_path)[0] + ".jpg"
    with open(cover_path, "wb") as f:
        f.write(requests.get(thumbnail).content)

# Создаем содержимое .nfo файла
nfo_content = f"""<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<movie>
  <title>{title}</title>
  <plot>{description}</plot>
  <thumb>{cover_path or "Нет обложки"}</thumb>
</movie>
"""

# Сохраняем .nfo файл
nfo_path = os.path.splitext(info_json_path)[0] + ".nfo"
with open(nfo_path, "w", encoding="utf-8") as nfo_file:
    nfo_file.write(nfo_content)

print(f".nfo файл успешно создан: {nfo_path}")
return nfo_path

# Основная программа
def main():
    url = "https://www.youtube.com/watch?v=HbsPPpeACvI"
    output_dir = "./downloads"
    cookies_file = "./cookies.txt" # Укажите путь к файлу cookies

    os.makedirs(output_dir, exist_ok=True)

    try:
        info_json_path = download_video(url, output_dir, cookies_file)
        create_nfo_file(info_json_path)
```

```
except Exception as e:  
    print(f"Произошла ошибка: {e}")
```

Ваш скрипт выглядит хорошо структурированным, но, похоже, в нем не хватает нескольких критических моментов для обеспечения правильной работы:

1. **Отсутствует импорт `requests`** : скрипт использует `requests.get` для загрузки миниатюры, но не импортирует библиотеку `requests` .
2. **Отладка загрузки видео**: если в выходном каталоге видео не отображается, это может быть связано со следующими причинами:
 - Неправильный файл cookie.
 - Недостаточные разрешения для команды `yt-dlp` .
3. **Обеспечение проверки файла метаданных**: проверка `.info.json` может завершиться неудачей, если каталог содержит несколько видео. Убедитесь, что вы проверяете правильный файл метаданных.

Вот исправленная и улучшенная версия скрипта:

Полный скрипт с корректировками

```
import sys  
import time  
sys.stdout.reconfigure(encoding='utf-8')  
  
import os  
import subprocess  
import json  
import requests # Ensure this is installed: `pip install requests`  
  
def download_video(url, output_dir, cookies_file):  
    # Путь для выходного видеофайла  
    output_template = os.path.join(output_dir, "%(title)s.%(ext)s")  
  
    # Команда yt-dlp для загрузки видео с использованием cookies  
    command = [
```

```
    "yt-dlp",
    "-f", "bestvideo+bestaudio/best", # Выбор лучшего видео и аудио
    "--merge-output-format", "mkv", # Сохранение в MKV
    "-o", output_template, # Шаблон имени файла
    "--write-info-json", # Сохранение метаданных
    "--cookies", cookies_file, # Передача cookies для аутентификации
    url
]

print("Загружается видео...")
try:
    subprocess.run(command, check=True)
    print("Видео успешно загружено.")
except subprocess.CalledProcessError as e:
    raise RuntimeError(f"Ошибка при загрузке видео: {e}")

# Поиск JSON файла с метаданными
info_json_path = None
for file in os.listdir(output_dir):
    if file.endswith(".info.json"):
        info_json_path = os.path.join(output_dir, file)
        break

if not info_json_path:
    raise FileNotFoundError("Файл метаданных .info.json не найден.")

return info_json_path

def create_nfo_file(info_json_path):
    # Читаем данные из JSON файла
    with open(info_json_path, "r", encoding="utf-8") as f:
        metadata = json.load(f)

    title = metadata.get("title", "Без названия")
    description = metadata.get("description", "Нет описания")
    thumbnail = metadata.get("thumbnail", "")

    # Загружаем обложку, если доступна
    cover_path = None
    if thumbnail:
        cover_path = os.path.splitext(info_json_path)[0] + ".jpg"
        try:
            response = requests.get(thumbnail)
```

```
        response.raise_for_status()
        with open(cover_path, "wb") as f:
            f.write(response.content)
        print(f"Обложка сохранена: {cover_path}")
    except Exception as e:
        print(f"Ошибка при загрузке обложки: {e}")
        cover_path = None

# Создаем содержимое .nfo файла
nfo_content = f"""<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<movie>
  <title>{title}</title>
  <plot>{description}</plot>
  <thumb>{cover_path or "Нет обложки"}</thumb>
</movie>
"""

# Сохраняем .nfo файл
nfo_path = os.path.splitext(info_json_path)[0] + ".nfo"
with open(nfo_path, "w", encoding="utf-8") as nfo_file:
    nfo_file.write(nfo_content)

print(f".nfo файл успешно создан: {nfo_path}")
return nfo_path

# Основная программа
def main():
    url = "https://www.youtube.com/watch?v=HbsPPpeACvI"
    output_dir = "./downloads"
    cookies_file = "./cookies.txt" # Укажите путь к файлу cookies

    os.makedirs(output_dir, exist_ok=True)

    try:
        # Загрузка видео и метаданных
        info_json_path = download_video(url, output_dir, cookies_file)

        # Создание .nfo файла
        create_nfo_file(info_json_path)
    except Exception as e:
        print(f"Произошла ошибка: {e}")
```

```
if __name__ == "__main__":  
    main()
```

Ключевые улучшения

1. Библиотека запросов:

- Добавлены необходимые `import requests`.
- Добавлена обработка ошибок при загрузке миниатюры.

2. Улучшенные сообщения об ошибках:

- Улучшены сообщения об исключениях для лучшей отладки.

3. Гарантированное соответствие файла метаданных:

- Теперь гарантируется явная привязка `.info.json` к загруженному видео.

4. Разрешения и зависимости:

- Убедитесь, что `yt-dlp` установлен и доступен в терминале. Запустите `pip install yt-dlp`.

Дополнительные примечания

- **Файл cookie:** Убедитесь, что файл `cookies.txt` правильно экспортирован из вашего браузера (выполнен вход в YouTube).
- **Отладка:** если видео не отображается в `./downloads`, убедитесь, что `yt-dlp` может быть запущен из терминала с помощью той же команды.
- **Журнал ошибок:** любые `RuntimeError` или отсутствующие файлы теперь будут вызывать описательные ошибки.

Дайте мне знать, если вам нужны дополнительные разъяснения!

Промт №5: Kodi не распознаёт описания в .nfo файле

У меня скачалось отдельно видео и отдельно звуковая дорожка:

```
[info] HbsPPpeACvI: Downloading 1 format(s): 616+251
[info] Writing video metadata as JSON to: downloads\????? ? ???? ???? ? ????
WARNING: You have requested merging of multiple formats but ffmpeg is not installed. Th
[hlsnative] Downloading m3u8 manifest
[hlsnative] Total fragments: 675
```

Проблема возникает из-за предупреждения о том, что `ffmpeg` не установлен. Без `ffmpeg` `yt-dlp` не может объединить отдельные видео- и аудиодорожки в один файл, что приводит к отдельным загрузкам для видео- и аудиопотоков.

Вот как решить проблему:

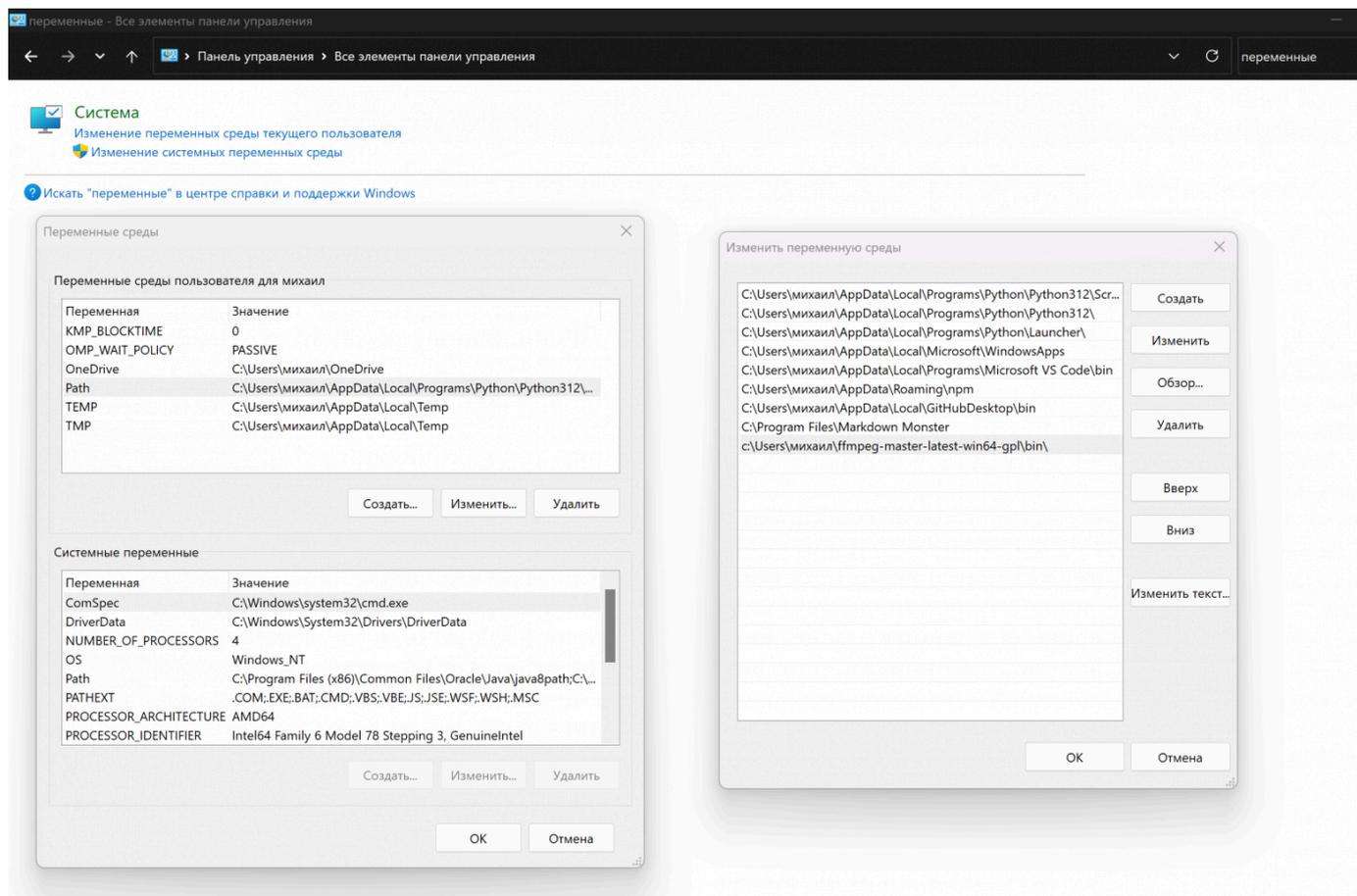
Шаги по устранению проблемы

1. Установите `ffmpeg` :

- `ffmpeg` требуется для объединения видео- и аудиодорожек.
- Вы можете установить его следующими способами в зависимости от вашей ОС:

В Windows:

- Загрузите последнюю сборку `ffmpeg` с официального сайта: ffmpeg.org.
- Извлеките загруженный файл и добавьте каталог `bin` в PATH вашей системы:
- Щелкните правой кнопкой мыши "Этот компьютер" > Свойства > Дополнительные параметры системы > Переменные среды.
- Отредактируйте переменную `Path` в "Системных переменных" и добавьте путь к папке `ffmpeg/bin`.



В Linux:

```
sudo apt update
sudo apt install ffmpeg
```

В macOS:

```
brew install ffmpeg
```

2. Проверьте установку `ffmpeg` :

- Откройте терминал или командную строку и выполните:

```
ffmpeg -version
```


Замените `video.mp4` и `audio.m4a` на фактические имена файлов ваших видео- и аудиодорожек.

Следующие шаги в вашем скрипте

После того, как вы убедитесь, что `ffmpeg` установлен:

- Видео и аудио будут автоматически объединены, когда вы используете команду `yt-dlp` с опцией `--merge-output-format` в вашем скрипте.

Дайте мне знать, если вам нужна помощь с установкой или тестированием!

Промт №6: заработало

Спасибо, всё заработало, но Kodi не распознаёт описания.

Для того чтобы Kodi распознавал обложку файл должен заканчиваться `-fanart.jpg`, а сейчас `.info.jpg`.

Для того чтобы Kodi распознавал описание файл должен заканчиваться `.nfo`, а сейчас `.info.nfo`.

Вот выходные данные скрипта:

```
[Merger] Merging formats into "downloads\One Track Mind - I Like You.mkv"
Deleting original file downloads\One Track Mind - I Like You.f134.mp4 (pass -k to keep)
Deleting original file downloads\One Track Mind - I Like You.f140.m4a (pass -k to keep)
Видео успешно загружено.
Обложка сохранена: ./downloads\One Track Mind - I Like You.info.jpg
.nfo файл успешно создан: ./downloads\One Track Mind - I Like You.info.nfo
```

Чтобы обеспечить совместимость с Kodi, нам нужно настроить скрипт так, чтобы изображение обложки и имена файлов `.nfo` соответствовали соглашениям Kodi. Вот обновленная версия скрипта:

Обновленный скрипт

```
import sys
import os
import subprocess
import json
import requests

sys.stdout.reconfigure(encoding="utf-8")

def download_video(url, output_dir, cookies_file):
    # Путь для выходного видеофайла
    output_template = os.path.join(output_dir, "%(title)s.%(ext)s")

    # Команда yt-dlp для загрузки видео с использованием cookies
    command = [
        "yt-dlp",
        "-f", "bestvideo+bestaudio/best", # Выбор лучшего видео и аудио
        "--merge-output-format", "mkv", # Сохранение в MKV
        "-o", output_template, # Шаблон имени файла
        "--write-info-json", # Сохранение метаданных
        "--cookies", cookies_file, # Передача cookies для аутентификации
        url
    ]

    print("Загружается видео...")
    subprocess.run(command, check=True)
    print("Видео успешно загружено.")

    # Находим JSON файл с метаданными
    info_json_path = next(
        (os.path.join(output_dir, f) for f in os.listdir(output_dir) if f.endswith(".info.json"))
    )
    if not info_json_path:
        raise FileNotFoundError("Файл метаданных .info.json не найден.")

    return info_json_path

def create_kodi_friendly_files(info_json_path):
    # Читаем данные из JSON файла
    with open(info_json_path, "r", encoding="utf-8") as f:
        metadata = json.load(f)
```

```
title = metadata.get("title", "Без названия")
description = metadata.get("description", "Нет описания")
thumbnail = metadata.get("thumbnail", "")

# Путь к базовым файлам (без расширений)
base_path = os.path.splitext(info_json_path)[0].replace(".info", "")

# Загружаем обложку, если доступна
cover_path = f"{base_path}-poster.jpg"
if thumbnail:
    with open(cover_path, "wb") as f:
        f.write(requests.get(thumbnail).content)
    print(f"Обложка сохранена: {cover_path}")
else:
    print("Обложка не найдена.")

# Создаем содержимое .nfo файла
nfo_content = f"""<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<movie>
  <title>{title}</title>
  <plot>{description}</plot>
  <thumb>{cover_path}</thumb>
</movie>
"""

# Сохраняем .nfo файл
nfo_path = f"{base_path}.nfo"
with open(nfo_path, "w", encoding="utf-8") as nfo_file:
    nfo_file.write(nfo_content)

print(f".nfo файл успешно создан: {nfo_path}")

# Основная программа
def main():
    url = "https://www.youtube.com/watch?v=HbsPPpeACvI"
    output_dir = "./downloads"
    cookies_file = "./cookies.txt" # Укажите путь к файлу cookies

    os.makedirs(output_dir, exist_ok=True)

    try:
```

```
info_json_path = download_video(url, output_dir, cookies_file)
create_kodi_friendly_files(info_json_path)
except Exception as e:
    print(f"Произошла ошибка: {e}")

if __name__ == "__main__":
    main()
```

Ключевые изменения

1. Переименование изображения обложки:

- Изображение обложки теперь сохраняется с суффиксом `-poster.jpg` (например, `One Track Mind - I Like You-poster.jpg`).

2. Переименование файла `.nfo` :

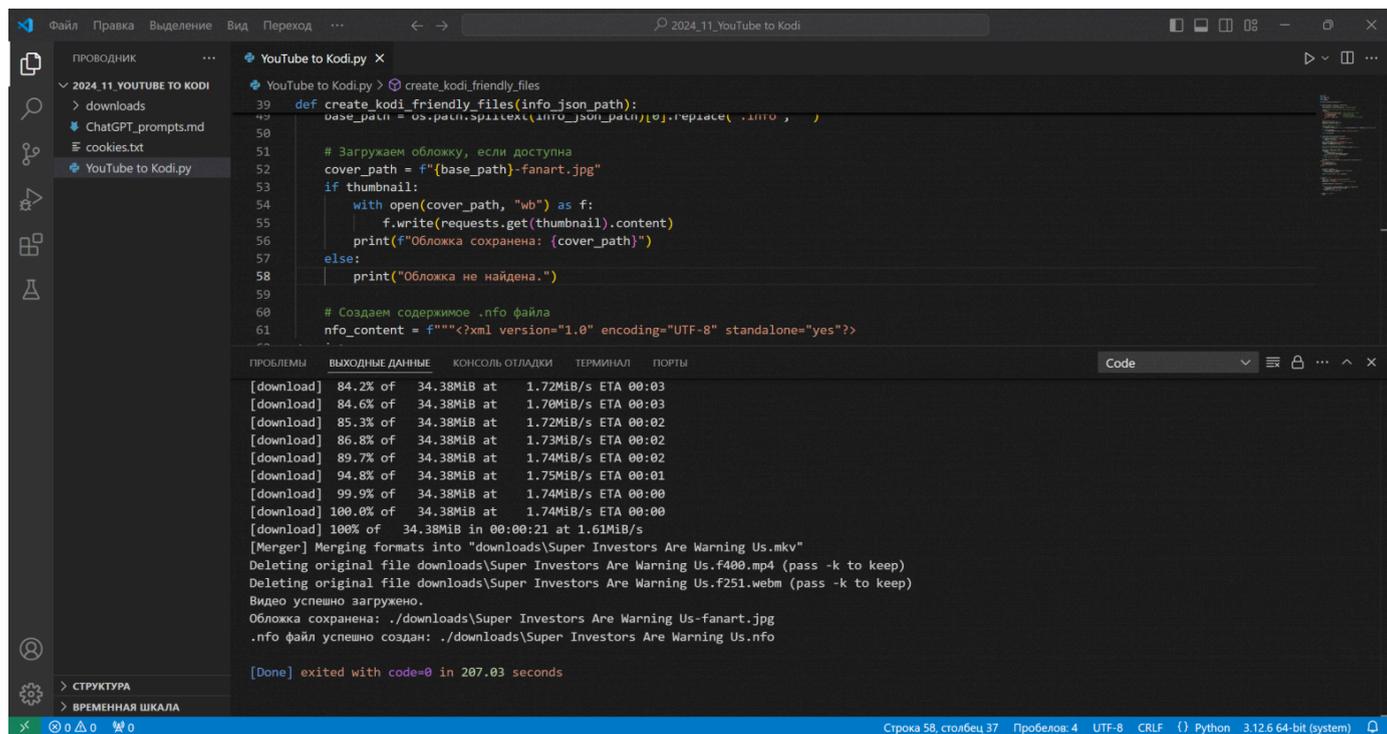
- Файл `.nfo` теперь соответствует имени видеофайла без суффикса `.info` (например, `One Track Mind - I Like You.nfo`).

Пример вывода

После запуска скрипта вы должны увидеть:

```
[Объединение] Объединение форматов в "downloads\One Track Mind - I Like You.mkv"
Удаление исходного файла downloads\One Track Mind - I Like You.f134.mp4 (передайте -k д
Удаление исходного файла downloads\One Track Mind - I Like You.f140.m4a (передайте -k д
Видео успешно загружено.
Обложка сохранена: downloads\One Track Mind - I Like You-poster.jpg
Файл .nfo успешно создан: downloads\One Track Mind - I Like You.nfo
```

Теперь Kodi должен распознать как метаданные видео, так и обложку.



```
YouTube to Kodi.py X
2024_11_YOUTUBE TO KODI
  downloads
  ChatGPT_prompts.md
  cookies.txt
  YouTube to Kodi.py

def create_kodi_friendly_files(info_json_path):
    base_path = os.path.splitext(info_json_path)[0].replace('_', '-')

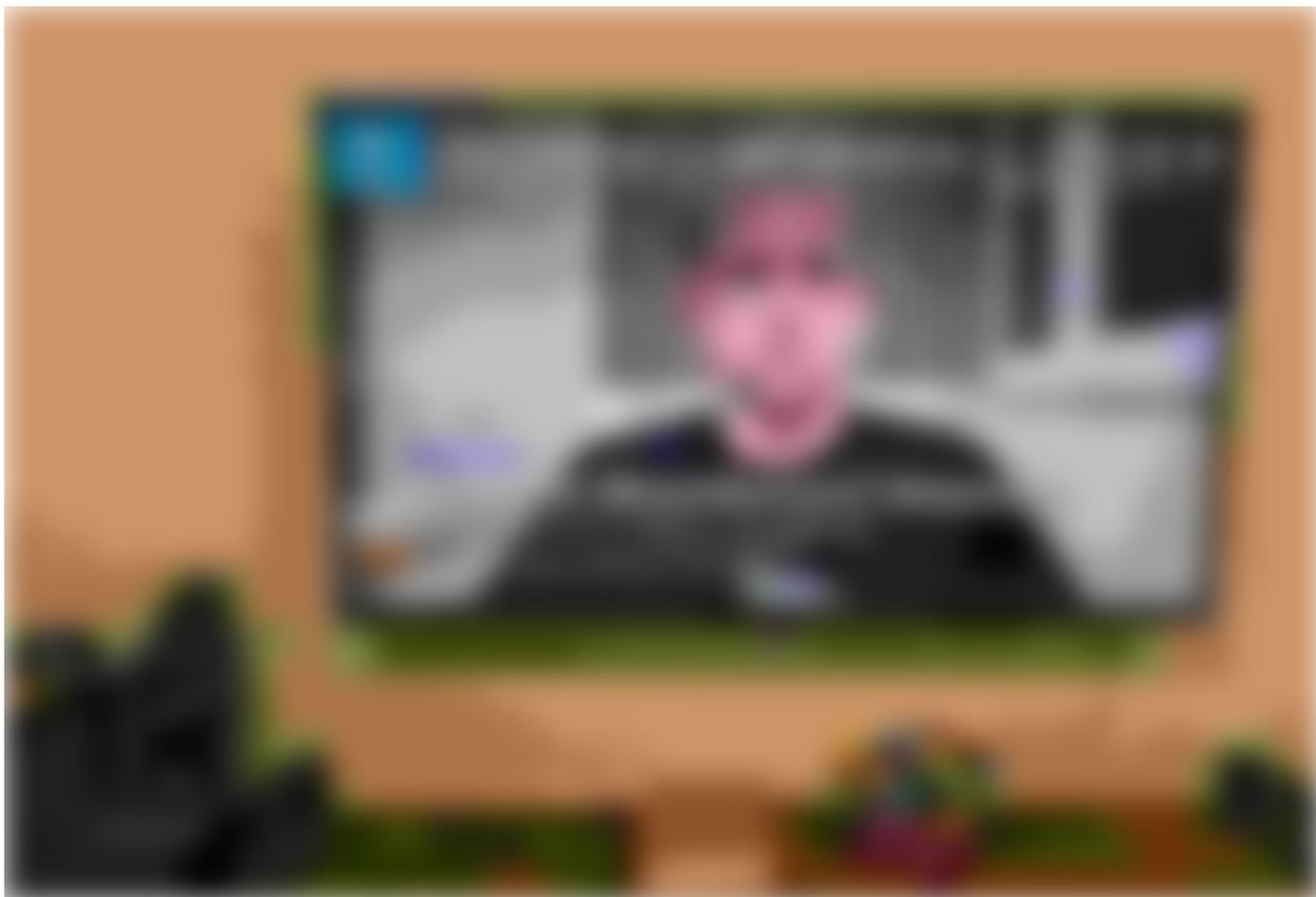
    # Загружаем обложку, если доступна
    cover_path = f"{base_path}-fanart.jpg"
    if thumbnail:
        with open(cover_path, "wb") as f:
            f.write(requests.get(thumbnail).content)
        print(f"Обложка сохранена: {cover_path}")
    else:
        print("Обложка не найдена.")

    # Создаем содержимое .nfo файла
    nfo_content = f"""<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

[download] 84.2% of 34.38MiB at 1.72MiB/s ETA 00:03
[download] 84.6% of 34.38MiB at 1.70MiB/s ETA 00:03
[download] 85.3% of 34.38MiB at 1.72MiB/s ETA 00:02
[download] 86.8% of 34.38MiB at 1.73MiB/s ETA 00:02
[download] 89.7% of 34.38MiB at 1.74MiB/s ETA 00:02
[download] 94.8% of 34.38MiB at 1.75MiB/s ETA 00:01
[download] 99.9% of 34.38MiB at 1.74MiB/s ETA 00:00
[download] 100.0% of 34.38MiB at 1.74MiB/s ETA 00:00
[download] 100% of 34.38MiB in 00:00:21 at 1.61MiB/s
[Merge] Merging formats into "downloads\Super Investors Are Warning Us.mkv"
Deleting original file downloads\Super Investors Are Warning Us.f400.mp4 (pass -k to keep)
Deleting original file downloads\Super Investors Are Warning Us.f251.webm (pass -k to keep)
Видео успешно загружено.
Обложка сохранена: ./downloads\Super Investors Are Warning Us-fanart.jpg
.nfo файл успешно создан: ./downloads\Super Investors Are Warning Us.nfo

[Done] exited with code=0 in 207.03 seconds
```

Итоговый вариант на GitHub



Подкаст на телевизоре

Выложил итоговый рабочий вариант на Гитхаб:

<https://github.com/empenoso/YouTube-to-Kodi/>

Итоги

Всего за 6 запросов и около получаса можно создать рабочий код, не углубляясь в подробности работы скрипта.

Автор: [Михаил Шардин](#)

25 ноября 2024 г.

Теги: [kodi](#), [youtube](#)

Хэбы: [Open source](#), [Умный дом](#), [Искусственный интеллект](#), [Python](#)

Редакторский дайджест



Присылаем лучшие статьи раз в месяц

**180**

Карма

33.5

Рейтинг

Михаил Шардин @empenoso

Разработчик

[Подписаться](#)[Сайт](#) [Сайт](#) [Github](#)

Комментарии 42

Публикации

ЛУЧШИЕ ЗА СУТКИ

ПОХОЖИЕ



ItwithMisha

21 час назад

Старики будут править IT



Простой



6 мин



43K

Мнение

+108

59

192



big-mdm

19 часов назад

Маркировка DC/DC-преобразователей в корпусе SOT-23-5 и SOT-23-6



Простой



3 мин



2.4K

Обзор

+55

54

14



k0mar0v

18 часов назад

Ретрокомпьютер моей мечты. Как я его собирал в 2025 году и что получилось в итоге



6 мин



7.1K

+46

15

27

**waffleglimmer**

23 часа назад

Как перенести Linux Device Drivers на современные ядра

🕒 13 мин

👁️ 3.3K

💎 +41

📖 53

💬 1

**ru_vds**

21 час назад

RustDesk: удалённый десктоп через свой сервер ретрансляции

📊 Средний

🕒 5 мин

👁️ 8.3K

Тutorial

💎 +32

📖 74

💬 18

**myoffice_ru**

15 часов назад

Самый странный лексический синтаксис, который я обнаружила, исследовав 42 языка программирования

🕒 13 мин

👁️ 9.5K

Обзор

Перевод

💎 +30

📖 38

💬 11

**Lexx_Nimoff**

17 часов назад

Почему игра «Сатурн» бесплатная, отчёты перед ИРИ и ждать ли мобильную версию. Интервью с пиар-директором игры

🗨️ Простой

🕒 9 мин

👁️ 2.9K

Интервью

💎 +28

📖 10

💬 4

**techno_mot**

21 час назад

Как Kubernetes стал стандартом управления инфраструктурой

🕒 5 мин 👁 4.8K

Обзор

📌 +26

📖 25

💬 5



EIKornacio

11 часов назад

Заменяем хабраюзеров ИИ-агентами. Гайд по browser-use

👉 Простой 🕒 4 мин 👁 2.1K

Обзор

📌 +21

📖 25

💬 22



ru_vds

17 часов назад

Станет ли ИИ катастрофой для сквозного шифрования?

📊 Средний 🕒 12 мин 👁 2.7K

Обзор

Перевод

📌 +20

📖 18

💬 9

Как закрыть ноут в 18:00 и встать со стула — комикс о переработках

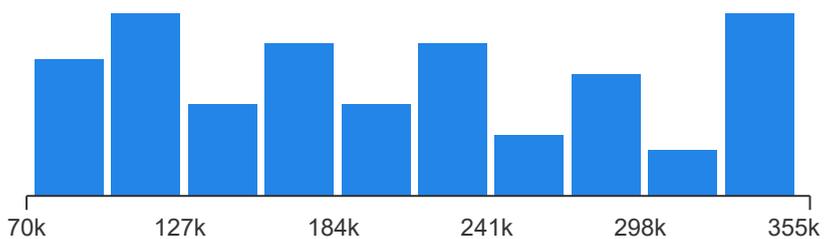
Промо

Показать еще

СРЕДНЯЯ ЗАРПЛАТА В IT

199 505 ₽/мес.

— средняя зарплата во всех IT-специализациях по данным из 11 048 анкет, за 1-ое пол. 2025 года. Проверьте «в рынке» ли ваша зарплата или нет!



Проверить свою зарплату

МИНУТОЧКУ ВНИМАНИЯ



Гиперконвергентная среда: OpenStack или VMware?



Техноархеолог отправляется в затерянный город и спасает мир



Печеньки, удаленка, обучение — а что важно в работе для вас?

РАБОТА

Django разработчик
18 вакансий

Python разработчик
72 вакансии

Data Scientist
53 вакансии

[Все вакансии](#)

БЛИЖАЙШИЕ СОБЫТИЯ



30 января

Зимний тест-драйв Хабра для компаний

Москва

Маркетинг

Другое

[Больше событий в календаре](#)

Хабр



🌐 [Настройка языка](#)

[Техническая поддержка](#)

