



YouTube Playlist Summary

Transforme playlists do YouTube em material de estudo estruturado.

Uma ferramenta automatizada que baixa vídeos, extrai legendas (nativas ou via IA) e gera material educacional consolidado — tudo em um único comando.



Propósito







Assistir horas de vídeos educacionais é demorado. Este projeto resolve esse problema ao:

1. **Baixar** vídeos ou áudios de playlists YouTube
2. **Obter legendas** automaticamente (prioriza legendas nativas; usa Whisper AI como fallback)
3. **Gerar material de estudo** consolidado via GPT — resumos, conceitos-chave, exemplos práticos e glossário

Resultado: Um documento Markdown completo que substitui a necessidade de assistir aos vídeos.



Principais Funcionalidades

Funcionalidade	Descrição
 Download inteligente	Baixa vídeos/áudios com controle de rate-limiting
 Legendas automáticas	Prioriza legendas do YouTube; usa Whisper AI se não disponíveis
 Checkpoint/Retomada	Interrompa e retome a qualquer momento (Ctrl+C seguro)
 Material de estudo	Gera documento educacional completo via GPT
 Multi-idioma inteligente	Detecta idioma do SO, seleciona legendas por prioridade, evita duplicatas
 Modo áudio	Opção para baixar apenas áudio (economia de espaço)



Pré-requisitos

- **Python** 3.10 ou superior
- **FFmpeg** e **ffprobe** instalados e no PATH
- **Chave API OpenAI** (para transcrição Whisper e geração de material)

Instalação do FFmpeg

Windows (via winget):

```
winget install FFmpeg.FFmpeg
```

Windows (via Chocolatey):

```
choco install ffmpeg
```

macOS:

```
brew install ffmpeg
```

Linux (Debian/Ubuntu):

```
sudo apt install ffmpeg
```



Instalação

1. Clone o repositório:

```
git clone https://github.com/seu-usuario/yt-playlist-summary.git  
cd yt-playlist-summary
```

2. Crie um ambiente virtual (recomendado):

```
python -m venv venv  
venv\Scripts\activate # Windows  
# ou  
source venv/bin/activate # Linux/macOS
```

3. Instale as dependências:

```
pip install -r requirements.txt
```

4. Configure a chave API:

```
# Opção 1: Variável de ambiente  
export OPENAI_API_KEY="sk-..." # Linux/macOS
```

```
set OPENAI_API_KEY=sk-...      # Windows CMD
$env:OPENAI_API_KEY="sk-..."  # Windows PowerShell

# Opção 2: Arquivo .env na raiz do projeto
echo OPENAI_API_KEY=sk-... > .env
```

Uso

Comando Básico

```
python yt_playlist_summary.py --url "URL_DA_PLAYLIST"
```

O que acontece por padrão:

1. ☒ Baixa todos os vídeos da playlist
2. ☒ Busca legendas nativas (pt-BR, en)
3. ☒ Se não encontrar legendas → transcreve via Whisper AI
4. ☒ Gera material de estudo consolidado
5. ☒ Checkpoint habilitado (pode interromper e retomar)

Exemplos Práticos

```
# Processar playlist completa (comportamento padrão)
python yt_playlist_summary.py --url "https://youtube.com/playlist?list=..."

# Modo interativo (confirma antes de cada etapa)
python yt_playlist_summary.py --url "URL" --interactive

# Apenas áudio (economia de espaço)
python yt_playlist_summary.py --url "URL" --audio-only

# Forçar uso do Whisper (ignorar legendas nativas)
python yt_playlist_summary.py --url "URL" --no-prefer-existing-subtitles

# Sem material de estudo (apenas download + legendas)
python yt_playlist_summary.py --url "URL" --no-study-material

# Limpar checkpoint e reprocessar tudo
python yt_playlist_summary.py --url "URL" --clear-checkpoint

# Especificar idioma fonte das legendas (prioridade)
python yt_playlist_summary.py --url "URL" --source-language pt-BR,en
```

```
# Material de estudo em inglês a partir de Legendas em português
python yt_playlist_summary.py --url "URL" --source-language pt-BR --study-

# Material em inglês usando Legendas em inglês
python yt_playlist_summary.py --url "URL" --source-language en --study-lan

# Material em português usando Legendas em inglês (tradução automática)
python yt_playlist_summary.py --url "URL" --source-language en --study-lan

# Forçar idioma específico (ignorar detecção do SO)
python yt_playlist_summary.py --url "URL" --source-language ja,en --study-
```

Estrutura de Saída

```
output/
├─ downloads/          # Vídeos/áudios originais
├─ audio/              # Áudio extraído (quando necessário)
├─ converted/          # Áudio 64kbps mono (para Whisper)
├─ subtitles/          # Arquivos .srt
├─ study_material_*.md # Material de estudo gerado
└─ .checkpoint_*.json  # Progresso (para retomada)
```

⚙️ Parâmetros Disponíveis



Parâmetro	Padrão	Descrição
-u, --url	<i>obrigatório</i>	URL da playlist ou vídeo
-k, --api-key	env OPENAI_API_KEY	Chave API OpenAI
-o, --output	./output	Diretório de saída
-l, --language	auto-detect	Idioma para transcrição Whisper
-a, --audio-only	False	Baixar apenas áudio
-i, --interactive	False	Modo interativo com confirmações
-v, --verbose	False	Logs detalhados
--subtitle-languages	pt-BR,en	Idiomas para buscar legendas
--download-delay	5	Segundos entre downloads

<code>--keep-original</code>	False	Manter áudio sem conversão
<code>--skip-transcription</code>	False	Pular etapa de legendas
<code>--no-prefer-existing-subtitles</code>	False	Forçar Whisper (ignorar legendas nativas)
<code>--no-study-material</code>	False	Não gerar material de estudo
<code>--source-language</code>	<i>idioma do SO</i>	Idioma(s) fonte das legendas (ex: pt-BR, en)
<code>--study-language</code>	<i>idioma do SO</i>	Idioma do material de saída
<code>--no-checkpoint</code>	False	Desabilitar checkpoint
<code>--clear-checkpoint</code>	False	Limpar checkpoint e reiniciar

Sistema de Checkpoint

O projeto salva progresso automaticamente. Se interromper (Ctrl+C), basta executar o mesmo comando novamente:

```
# Primeira execução - interrompida no vídeo 5/20
python yt_playlist_summary.py --url "URL"
# ^C

# Segunda execução - retoma do vídeo 6
python yt_playlist_summary.py --url "URL"
#  RETOMANDO DOWNLOAD
#  Já concluídos: 5/20
```

Scripts Auxiliares

Traduzir legendas existentes

```
python translate_sub.py \
  --input ./output/subtitles/video.pt-BR.srt \
  --source pt-BR \
  --target en
```

Gerar material de estudo a partir de legendas prontas

```
# Usar padrões do sistema (detecta idioma do SO)
python generate_study_material.py -s ./output/subtitles

# Especificar idioma fonte e de saída
python generate_study_material.py \
  --subtitle-dir ./output/subtitles \
  --source-language pt-BR,en \
  --output-language pt

# Modo interativo (pergunta idiomas)
python generate_study_material.py -s ./output/subtitles -i

# Apenas consolidar (sem GPT)
python generate_study_material.py -s ./output/subtitles --skip-gpt
```

Transcrever arquivo de áudio isolado

```
python mywhisper.py --input audio.mp3
```

Renomear arquivos usando checkpoint

```
python rename_from_checkpoint.py \
  --checkpoint output/.checkpoint_abc123.json
```

Estimativa de Custos (OpenAI)

Operação	Custo Aproximado
Whisper (transcrição)	~\$0.006 por minuto de áudio
GPT (material de estudo)	~\$0.02-0.05 por playlist típica (5-10 vídeos)

Dica: Use `--prefer-existing-subtitles` (padrão) para economizar — legendas nativas são gratuitas!

Arquitetura do Projeto

```
yt-playlist-summary/
├─ yt_playlist_summary.py  # Orquestrador principal do pipeline
├─ mywhisper.py           # Transcrição via Whisper + cache
```

```
|— generate_study_material.py # Geração de material educacional
|— language_utils.py         # Detecção de idioma do SO e seleção intelig
|— checkpoint_manager.py     # Sistema de checkpoint/retomada
|— translate_sub.py          # Tradução de SRT via GPT
|— rename_from_checkpoint.py # Utilitário de renomeação
|— requirements.txt          # Dependências Python
|— README.md
```

Seleção Inteligente de Idiomas

O sistema detecta automaticamente o idioma do seu sistema operacional e configura os padrões:

SO em Português	SO em Inglês
Fonte: pt-BR, pt, und	Fonte: en-US, en, und
Saída: pt	Saída: en

Como funciona

1. **Agrupar legendas por vídeo** — identifica índice pelo nome do arquivo
2. **Seleciona uma legenda por vídeo** — usa a prioridade de idiomas configurada
3. **Evita duplicatas** — economiza tokens do GPT!

Exemplo prático:

```
Subtitles/
|— 1. Intro.en.srt
|— 1. Intro.pt-BR.srt ← selecionado (pt-BR tem prioridade)
|— 2. Review.en.srt
|— 2. Review.pt-BR.srt ← selecionado
```

Resultado: 2 legendas processadas em vez de 4!

Códigos de idioma suportados (BCP 47)

pt, pt-BR, en, en-US, es, fr, de, it, ja, zh, ko, ru, ar, hi

Solução de Problemas

Problema	Solução
FFmpeg not found	Instale FFmpeg e adicione ao PATH

API key not found	Configure OPENAI_API_KEY via env ou --api-key
Erro de rate-limiting	Aumente --download-delay (ex: 10 ou 15)
Vídeo privado/indisponível	O script pula automaticamente e continua
Checkpoint corrompido	Use --clear-checkpoint para reiniciar

Licença

MIT License - Veja [LICENSE](#) para detalhes.

Contribuições

Contribuições são bem-vindas! Por favor, mantenha a separação de responsabilidades:

- yt_playlist_summary.py → download e pré-processamento
 - mywhisper.py → transcrição e manipulação de legendas
 - Novos módulos → funcionalidades independentes
-

Feito com  para tornar o aprendizado mais eficiente.