yt-dlp / yt-dlp Публичный

A feature-rich command-line audio/video downloader

discord.gg/h5mncfw63r

Unlicense license

78.1k stars 6.1k forks Branches Tags Activity

Star Notifications

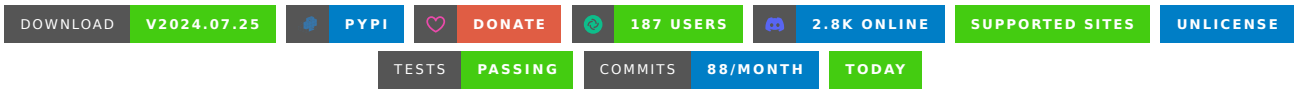
Код Проблемы 1.4k Запросы на извлечение 227 Обсуждения Действия Проекты 4 Неделя Безопасность 6

владелец 3 отделения 93 Теги Перейти к файлу Go to file Код

башонли и Grub4K 5 hours ago

.github	Вернуть 4f84488	3 недели назад
пучок	[сборка] Включить curl_cffibyt-dlp_linux	3 недели назад
devscripts	[очистка] Разное (#10383)	3 недели назад
тест	[utils] unified_timestamp: Распознать воскре···	вчера
yt_dlp	[ie/learningonscreen] Добавить экстрактор···	5 часов назад
.editorconfig	[docs] Добавить .editorconfig файл (#3220)	2 года назад
.gitattributes	[документы] Незначительные улучшения (#3···	2 года назад
.gitignore	[очистка] Разное (#10330)	28 дней назад
.pre-commit-config.yaml	[разное] Добавить hatch, ruffи pre-commit ул···	2 месяца назад
.pre-commit-hatch.yaml	[разное] Добавить hatch, ruffи pre-commit ул···	2 месяца назад
ВКЛАД.md	[очистка] Разное (#10075)	29 дней назад
УЧАСТНИКИ	Выпуск 2024.07.16	2 недели назад
Changelog.md	Выпуск 2024.07.25	5 дней назад
Коллабораторы.md	[очистка] Разное (#10075)	29 дней назад
ЛИЦЕНЗИЯ	добавлено серьезное посвящение Public Dom···	12 лет назад
Makefile	[очистка] Разное (#10330)	28 дней назад
README.md	Выпуск 2024.07.16	2 недели назад
публичный.ключ	[сборка] Подписание файлов SHA и выпуск о···	в прошлом году
pyproject.toml	[сборка] setuptools Версия Pin (#10493)	2 недели назад
настройка.cfg	[разное] Добавить hatch, ruffи pre-commit ул···	2 месяца назад
supportedsites.md	Выпуск 2024.07.16	2 недели назад
yt-dlp.cmd	[очистка] Разное (#8598)	7 месяцев назад
yt-dlp.sh	[очистка] Разное (#8598)	7 месяцев назад

YT-DLP *A feature-rich command-line audio/video downloader*



yt-dlp — это многофункциональный командный загрузчик аудио/видео с поддержкой тысяч сайтов . Проект представляет собой ответвление youtube-dl, основанное на ныне неактивном youtube-dlc .

МОНТАЖ

Вы можете установить yt-dlp с помощью бинарных файлов , pip или с помощью стороннего менеджера пакетов. Подробные инструкции см . в вики

ФАЙЛЫ РЕЛИЗА

рекомендуемые

Файл	Описание
yt-dlp	Независимый от платформы двоичный файл zipimport . Требуется Python (рекомендуется для Linux/BSD)
yt-dlp.exe	Windows (Win7 SP1+) автономный двоичный файл x64 (рекомендуется для Windows)
yt-dlp_macos	Универсальный автономный исполняемый файл MacOS (10.15+) (рекомендуется для MacOS)

Альтернативы

Файл	Описание
yt-dlp_x86.exe	Windows (Win7 SP1+) автономный x86 (32-бит) двоичный
yt-dlp_min.exe	Автономный двоичный файл x64 для Windows (Win7 SP1+), собранный с помощью py2exe (не рекомендуется)
yt-dlp_linux	Автономный двоичный файл Linux x64
yt-dlp_linux_armv7l	Автономный двоичный файл armv7l (32-бит) для Linux
yt-dlp_linux_aarch64	Автономный двоичный файл Linux aarch64 (64-бит)
yt-dlp_win.zip	Неупакованный исполняемый файл Windows (без автоматического обновления)
yt-dlp_macos.zip	Распакованный исполняемый файл MacOS (10.15+) (без автоматического обновления)
yt-dlp_macos_legacy	Автономный исполняемый файл x64 для MacOS (10.9+)

Разное

Файл	Описание
yt-dlp.tar.gz	Исходный tar-архив
SHA2-512СУМЫ	Суммы SHA512 в стиле GNU
SHA2-512SUMS.сиг	Файл подписи GPG для сумм SHA512
SHA2-256СУМС	Суммы SHA256 в стиле GNU
SHA2-256SUMS.сиг	Файл подписи GPG для сумм SHA256

Открытый ключ, который можно использовать для проверки подписей GPG, доступен здесь. Пример использования:

```
curl -L https://github.com/yt-dlp/yt-dlp/raw/master/public.key | gpg --import
gpg --verify SHA2-256SUMS.sig SHA2-256SUMS
gpg --verify SHA2-512SUMS.sig SHA2-512SUMS
```

Примечание : страницы руководства, файлы автозаполнения оболочки и т. д. доступны внутри исходного tar-архива.

ОБНОВЛЯТЬ

Вы можете использовать `yt-dlp -U` для обновления, если вы используете релизные двоичные файлы

Если вы устанавливали с помощью pip , просто повторно запустите ту же команду, которая использовалась для установки программы.

Информацию о других сторонних менеджерах пакетов см. в вики или в их документации.

В настоящее время существует три канала выпуска двоичных файлов: `stable` , `nightly` и `master` .

- `stable` является каналом по умолчанию, и многие из его изменений были протестированы пользователями каналов `nightly` и `master` .
- Канал `nightly` имеет релизы, запланированные для сборки каждый день около полуночи UTC, для моментального снимка новых патчей и изменений проекта. Это рекомендуемый канал для постоянных пользователей yt-dlp. `nightly` Релизы доступны из yt-dlp/yt-dlp-nightly-builds или как релизы разработки пакета `yt-dlp` PyPI (который можно установить с помощью флага `pip --pre`).
- Канал `master` содержит релизы, которые собираются после каждого push в основную ветку, и они будут иметь самые последние исправления и дополнения, но также могут быть более подвержены регрессиям. Они доступны из yt-dlp/yt-dlp-master-builds .

При использовании `--update` / `-U` двоичный файл релиза будет обновляться только до текущего канала. `--update-to CHANNEL` может использоваться для переключения на другой канал при наличии более новой версии. `--update-to [CHANNEL@]TAG` также может использоваться для обновления или понижения уровня до определенных тегов из канала.

Вы также можете использовать `--update-to <repository>` (`<owner>/<repository>`) для обновления на канале в совершенно другом репозитории. Будьте осторожны с тем, на какой репозиторий вы обновляетесь, для двоичных файлов из разных репозиториев проверка не производится.

Пример использования:

- `yt-dlp --update-to master` переключитесь на `master` канал и обновитесь до последнего выпуска
- `yt-dlp --update-to stable@2023.07.06` обновление/понижение версии до версии с `stable` тегом канала `2023.07.06`
- `yt-dlp --update-to 2023.10.07` обновить/понизить тег `2023.10.07` , если он существует на текущем канале
- `yt-dlp --update-to example/yt-dlp@2023.09.24` обновление/понижение версии до версии из `example/yt-dlp` репозитория, тег `2023.09.24`

Важно : любой пользователь, столкнувшийся с проблемой в `stable` выпуске, должен установить или обновить его до `nightly` версии, прежде чем отправлять отчет об ошибке:

```
# To update to nightly from stable executable/binary:
yt-dlp --update-to nightly

# To install nightly with pip:
python3 -m pip install -U --pre "yt-dlp[default]"
```

ЗАВИСИМОСТИ

Поддерживаются версии Python 3.8+ (CPython и PyPy). Другие версии и реализации могут работать или не работать правильно.

В то время как все остальные зависимости являются необязательными `ffmpeg` и `ffprobe` настоятельно рекомендуются

Настоятельно рекомендуется

- ffmpeg и ffprobe - Необходимы для слияния отдельных видео и аудио файлов , а также для различных задач постобработки . Лицензия зависит от сборки

  В ffmpeg есть ошибки, которые вызывают различные проблемы при использовании вместе с yt-dlp. Поскольку ffmpeg является такой важной зависимостью, мы предоставляем пользовательские сборки с исправлениями для некоторых из этих проблем на yt-dlp/FFmpeg-Builds . Подробности о конкретных проблемах, решенных этими сборками, см. в readme

  Важно : Вам нужен именно *двоичный* файл `ffmpeg` , а НЕ пакет Python с таким же названием.

Нетворкинг

- certifi * - Предоставляет пакет корневых сертификатов Mozilla. Лицензия MPLv2
- brotli * или brotlicffi - Поддержка кодирования контента Brotli . Обе лицензии MIT[1] [2]
- websockets * - Для загрузки через websocket. Лицензия BSD-3-Clause

- запросы * - HTTP библиотека. Для поддержки HTTPS proxy и постоянных соединений. Лицензия Apache-2.0

Олицетворение

Ниже представлена поддержка для имитации запросов браузера. Это может потребоваться для некоторых сайтов, использующих TLS-отпечатки.

- curl_cffi (рекомендуется) — привязка Python к curl-impersonate . Предоставляет цели олицетворения для Chrome, Edge и Safari. Лицензируется MIT
  - Может быть установлен вместе с `curl-cffi` группой, например `pip install "yt-dlp[default,curl-cffi]"`
  - В настоящее время включен в `yt-dlp.exe` , `yt-dlp_linux` и `yt-dlp_macos` сборки

Метаданные

- мутаген * - Для `--embed-thumbnail` определенных форматов. Лицензия GPLv2+
- AtomicParsley - Для `--embed-thumbnail` файлов `mp4` / `m4a` , когда `mutagen` / `ffmpeg` не может. Лицензия GPLv2+
- xattr , pyxattr или setfattr - Для записи метаданных xattr (`--xattr`) на Mac и BSD . Лицензируется под MIT , LGPL2.1 и GPLv2+ соответственно

Разное

- pycryptodomex * - Для расшифровки потоков AES-128 HLS и различных других данных. Лицензия BSD-2-Clause
- phantomjs - Используется в экстракторах, где необходимо запустить javascript. Лицензируется в соответствии с BSD-3-Clause
- secretstorage * - Для `--cookies-from-browser` доступа к связке ключей Gnome при расшифровке файлов cookie браузеров на базе Chromium в Linux . Лицензия BSD-3-Clause
- Любой внешний загрузчик, который вы хотите использовать `--downloader`

Устаревший

- avconv и avprobe - Теперь устаревшая альтернатива ffmpeg. Лицензия зависит от сборки
- sponskrub - Для использования устаревших опций sponskrub . Лицензия GPLv3+
- rtmpdump - Для загрузки `rtmp` потоков. Вместо этого можно использовать `ffmpeg` с `--downloader ffmpeg` . Лицензия GPLv2+
- mplayer или mpv - Для загрузки `rstp` / `mms` потоковой передачи. Вместо этого можно использовать `ffmpeg` с `--downloader ffmpeg` . Лицензия GPLv2+

Чтобы использовать или распространять зависимости, вы должны согласиться с соответствующими условиями лицензирования.

Отдельные двоичные файлы выпуска собираются с использованием интерпретатора Python и пакетов, отмеченных знаком * .

Если у вас нет необходимых зависимостей для задачи, которую вы пытаетесь выполнить, yt-dlp предупредит вас. Все доступные в настоящее время зависимости видны в верхней части `--verbose` вывода

КОМПИЛЯЦИЯ

Автономные сборки PyInstaller

Для сборки автономного исполняемого файла вам необходимо иметь Python и `pyinstaller` (плюс любые необязательные зависимости yt-dlp , если необходимо). Исполняемый файл будет собран для той же архитектуры ЦП, что и используемый Python.

Вы можете выполнить следующие команды:

```
python3 devscripts/install_deps.py --include pyinstaller
python3 devscripts/make_lazy_extractors.py
python3 -m bundle.pyinstaller
```

В некоторых системах вам может потребоваться использовать `py` или `python` вместо `python3` .

`python -m bundle.pyinstaller` принимает любые аргументы, которые могут быть переданы `pyinstaller` , например , `--onefile/-F` или `--onedir/-D` , что более подробно описано здесь .

Примечание : версии Pyinstaller ниже 4.4 не поддерживают Python, установленный из Магазина Windows без использования виртуальной среды.

Важно : Запуск `pyinstaller` напрямую вместо использования официально `python -m bundle.pyinstaller` не поддерживается . Это может работать правильно, а может и нет.

Независимый от платформы двоичный файл (UNIX)

Вам понадобятся инструменты сборки `python` (3.8+), `zip`, `make` (GNU), `pandoc` * и `pytest` *.

После установки просто запустите `make`.

Вместо этого вы также можете запустить команду `make yt-dlp` скомпилировать только двоичный файл, не обновляя никаких дополнительных файлов. (Инструменты сборки, отмеченные *, для этого не нужны)

Автономные сборки Py2Exe (Windows)

Хотя мы предоставляем возможность сборки с помощью py2exe, рекомендуется выполнять сборку с помощью PyInstaller, поскольку сборки py2exe не могут содержать **`pycryptodomex`** // и для их запуска требуется VC++14 **`certifi requests`** на целевом компьютере.

Если вы все равно хотите его собрать, установите Python (если он еще не установлен) и выполните следующие команды:

```
py devscripts/install_deps.py --include py2exe
py devscripts/make_lazy_extractors.py
py -m bundle.py2exe
```

Похожие сценарии

- **`devscripts/install_deps.py`** - Установить зависимости для yt-dlp.
- **`devscripts/update-version.py`** - Обновите номер версии на основе текущей даты.
- **`devscripts/set-variant.py`** - Установите вариант сборки исполняемого файла.
- **`devscripts/make_changelog.py`** - Создайте журнал изменений в формате Markdown, используя короткие сообщения о коммитах и `CONTRIBUTORS` файл обновления.
- **`devscripts/make_lazy_extractors.py`** - Создайте ленивые экстракторы. Запуск этого перед сборкой двоичных файлов (любой вариант) улучшит их производительность при запуске. Установите переменную окружения, `YTDLP_NO_LAZY_EXTRACTORS=1` если вы хотите принудительно отключить загрузку ленивого экстрактора.

`--help` Примечание: более подробную информацию см. здесь.

Форкинг проекта

Если вы разветвляете проект на GitHub, вы можете запустить рабочий процесс сборки вашего форка, чтобы автоматически собрать выбранные версии как артефакты. В качестве альтернативы вы можете запустить рабочий процесс релиза или включить ночной рабочий процесс для создания полных (предварительных) релизов.

ИСПОЛЬЗОВАНИЕ И ВАРИАНТЫ

```
yt-dlp [OPTIONS] [--] URL [URL...]
```

`Ctrl+F` твой друг :D

Общие настройки:

```
-h, --help                      Print this help text and exit
--version                       Print program version and exit
-U, --update                    Update this program to the latest version
--no-update                     Do not check for updates (default)
--update-to [CHANNEL]@[TAG]     Upgrade/downgrade to a specific version.
                                CHANNEL can be a repository as well. CHANNEL
                                and TAG default to "stable" and "latest"
                                respectively if omitted; See "UPDATE" for
                                details. Supported channels: stable,
                                nightly, master
-i, --ignore-errors             Ignore download and postprocessing errors.
                                The download will be considered successful
                                even if the postprocessing fails
--no-abort-on-error             Continue with next video on download errors;
                                e.g. to skip unavailable videos in a
                                playlist (default)
--abort-on-error                Abort downloading of further videos if an
                                error occurs (Alias: --no-ignore-errors)
--dump-user-agent               Display the current user-agent and exit
--list-extractors               List all supported extractors and exit
--extractor-descriptions        Output descriptions of all supported
                                extractors and exit
--use-extractors NAMES          Extractor names to use separated by commas.
```

```
                                You can also use regexes, "all", "default"
                                and "end" (end URL matching); e.g. --ies
                                "holodex.*,end,youtube". Prefix the name
                                with a "-" to exclude it, e.g. --ies
                                default,-generic. Use --list-extractors for
                                a list of extractor names. (Alias: --ies)
    --default-search PREFIX     Use this prefix for unqualified URLs. E.g.
                                "gvsearch2:python" downloads two videos from
                                google videos for the search term "python".
                                Use the value "auto" to let yt-dlp guess
                                ("auto_warning" to emit a warning when
                                guessing). "error" just throws an error. The
                                default value "fixup_error" repairs broken
                                URLs, but emits an error if this is not
                                possible instead of searching
    --ignore-config             Don't load any more configuration files
                                except those given to --config-locations.
                                For backward compatibility, if this option
                                is found inside the system configuration
                                file, the user configuration is not loaded.
                                (Alias: --no-config)
    --no-config-locations       Do not load any custom configuration files
                                (default). When given inside a configuration
                                file, ignore all previous --config-locations
                                defined in the current file
    --config-locations PATH     Location of the main configuration file;
                                either the path to the config or its
                                containing directory ("-" for stdin). Can be
                                used multiple times and inside other
                                configuration files
    --flat-playlist             Do not extract the videos of a playlist,
                                only list them
    --no-flat-playlist          Fully extract the videos of a playlist
                                (default)
    --live-from-start           Download livestreams from the start.
                                Currently only supported for YouTube
                                (Experimental)
    --no-live-from-start        Download livestreams from the current time
                                (default)
    --wait-for-video MIN[-MAX]  Wait for scheduled streams to become
                                available. Pass the minimum number of
                                seconds (or range) to wait between retries
    --no-wait-for-video         Do not wait for scheduled streams (default)
    --mark-watched              Mark videos watched (even with --simulate)
    --no-mark-watched           Do not mark videos watched (default)
    --color [STREAM:]POLICY     Whether to emit color codes in output,
                                optionally prefixed by the STREAM (stdout or
                                stderr) to apply the setting to. Can be one
                                of "always", "auto" (default), "never", or
                                "no_color" (use non color terminal
                                sequences). Use "auto-tty" or "no_color-tty"
                                to decide based on terminal support only.
                                Can be used multiple times
    --compat-options OPTS       Options that can help keep compatibility
                                with youtube-dl or youtube-dlc
                                configurations by reverting some of the
                                changes made in yt-dlp. See "Differences in
                                default behavior" for details
    --alias ALIASES OPTIONS     Create aliases for an option string. Unless
                                an alias starts with a dash "-", it is
                                prefixed with "--". Arguments are parsed
                                according to the Python string formatting
                                mini-language. E.g. --alias get-audio,-X
                                "-S=aext:{0},abr -x --audio-format {0}"
                                creates options "--get-audio" and "-X" that
                                takes an argument (ARG0) and expands to
                                "-S=aext:ARG0,abr -x --audio-format ARG0".
                                All defined aliases are listed in the --help
                                output. Alias options can trigger more
                                aliases; so be careful to avoid defining
                                recursive options. As a safety measure, each
                                alias may be triggered a maximum of 100
                                times. This option can be used multiple times
```

Параметры сети:

```
--proxy URL                     Use the specified HTTP/HTTPS/SOCKS proxy. To
                                enable SOCKS proxy, specify a proper scheme,
                                e.g. socks5://user:pass@127.0.0.1:1080/.
                                Pass in an empty string (--proxy "") for
                                direct connection
--socket-timeout SECONDS        Time to wait before giving up, in seconds
--source-address IP             Client-side IP address to bind to
--impersonate CLIENT[:OS]       Client to impersonate for requests. E.g.
                                chrome, chrome-110, chrome:windows-10. Pass
                                --impersonate="" to impersonate any client.
                                Note that forcing impersonation for all
                                requests may have a detrimental impact on
                                download speed and stability
--list-impersonate-targets      List available clients to impersonate.
-4, --force-ipv4                Make all connections via IPv4
-6, --force-ipv6                Make all connections via IPv6
--enable-file-urls              Enable file:// URLs. This is disabled by
                                default for security reasons.
```

Гео-ограничение:

```
--geo-verification-proxy URL    Use this proxy to verify the IP address for
                                some geo-restricted sites. The default proxy
                                specified by --proxy (or none, if the option
                                is not present) is used for the actual
                                downloading
--xff VALUE                     How to fake X-Forwarded-For HTTP header to
                                try bypassing geographic restriction. One of
                                "default" (only when known to be useful),
                                "never", an IP block in CIDR notation, or a
                                two-letter ISO 3166-2 country code
```

Выбор видео:

```
-I, --playlist-items ITEM_SPEC  Comma separated playlist_index of the items
                                to download. You can specify a range using
                                "[START]:[STOP][:STEP]". For backward
                                compatibility, START-STOP is also supported.
                                Use negative indices to count from the right
                                and negative STEP to download in reverse
                                order. E.g. "-I 1:3,7,-5::2" used on a
                                playlist of size 15 will download the items
                                at index 1,2,3,7,11,13,15
--min-filesize SIZE             Abort download if filesize is smaller than
                                SIZE, e.g. 50k or 44.6M
--max-filesize SIZE             Abort download if filesize is larger than
                                SIZE, e.g. 50k or 44.6M
--date DATE                     Download only videos uploaded on this date.
                                The date can be "YYYYMMDD" or in the format
                                [now|today|yesterday][-N[day|week|month|year]].
                                E.g. "--date today-2weeks" downloads only
                                videos uploaded on the same day two weeks ago
--datebefore DATE               Download only videos uploaded on or before
                                this date. The date formats accepted is the
                                same as --date
--dateafter DATE                Download only videos uploaded on or after
                                this date. The date formats accepted is the
                                same as --date
--match-filters FILTER          Generic video filter. Any "OUTPUT TEMPLATE"
                                field can be compared with a number or a
                                string using the operators defined in
                                "Filtering Formats". You can also simply
                                specify a field to match if the field is
                                present, use "!field" to check if the field
                                is not present, and "&" to check multiple
                                conditions. Use a "\" to escape "&" or
                                quotes if needed. If used multiple times,
                                the filter matches if at least one of the
                                conditions is met. E.g. --match-filter
```

```
                                !is_live --match-filter "like_count>?100 &
                                description~='(?i)\bcats \& dogs\b'" matches
                                only videos that are not live OR those that
                                have a like count more than 100 (or the like
                                field is not available) and also has a
                                description that contains the phrase "cats &
                                dogs" (caseless). Use "--match-filter -" to
                                interactively ask whether to download each
                                video
--no-match-filters              Do not use any --match-filter (default)
--break-match-filters FILTER    Same as "--match-filters" but stops the
                                download process when a video is rejected
--no-break-match-filters        Do not use any --break-match-filters (default)
--no-playlist                   Download only the video, if the URL refers
                                to a video and a playlist
--yes-playlist                  Download the playlist, if the URL refers to
                                a video and a playlist
--age-limit YEARS               Download only videos suitable for the given
                                age
--download-archive FILE         Download only videos not listed in the
                                archive file. Record the IDs of all
                                downloaded videos in it
--no-download-archive           Do not use archive file (default)
--max-downloads NUMBER          Abort after downloading NUMBER files
--break-on-existing             Stop the download process when encountering
                                a file that is in the archive
--no-break-on-existing          Do not stop the download process when
                                encountering a file that is in the archive
                                (default)
--break-per-input               Alters --max-downloads, --break-on-existing,
                                --break-match-filter, and autonumber to
                                reset per input URL
--no-break-per-input            --break-on-existing and similar options
                                terminates the entire download queue
--skip-playlist-after-errors N  Number of allowed failures until the rest of
                                the playlist is skipped
```

Варианты загрузки:

```
-N, --concurrent-fragments N    Number of fragments of a dash/hlsnative
                                video that should be downloaded concurrently
                                (default is 1)
-r, --limit-rate RATE           Maximum download rate in bytes per second,
                                e.g. 50K or 4.2M
--throttled-rate RATE           Minimum download rate in bytes per second
                                below which throttling is assumed and the
                                video data is re-extracted, e.g. 100K
-R, --retries RETRIES           Number of retries (default is 10), or
                                "infinite"
--file-access-retries RETRIES   Number of times to retry on file access
                                error (default is 3), or "infinite"
--fragment-retries RETRIES      Number of retries for a fragment (default is
                                10), or "infinite" (DASH, hlsnative and ISM)
--retry-sleep [TYPE:]EXPR       Time to sleep between retries in seconds
                                (optionally) prefixed by the type of retry
                                (http (default), fragment, file_access,
                                extractor) to apply the sleep to. EXPR can
                                be a number, linear=START[:END[:STEP=1]] or
                                exp=START[:END[:BASE=2]]. This option can be
                                used multiple times to set the sleep for the
                                different retry types, e.g. --retry-sleep
                                linear=1::2 --retry-sleep fragment:exp=1:20
--skip-unavailable-fragments    Skip unavailable fragments for DASH,
                                hlsnative and ISM downloads (default)
                                (Alias: --no-abort-on-unavailable-fragments)
--abort-on-unavailable-fragments
                                Abort download if a fragment is unavailable
                                (Alias: --no-skip-unavailable-fragments)
--keep-fragments                Keep downloaded fragments on disk after
                                downloading is finished
--no-keep-fragments             Delete downloaded fragments after
                                downloading is finished (default)
--buffer-size SIZE              Size of download buffer, e.g. 1024 or 16K
                                (default is 1024)
--resize-buffer                 The buffer size is automatically resized
                                from an initial value of --buffer-size
```

```
                                (default)
    --no-resize-buffer          Do not automatically adjust the buffer size
    --http-chunk-size SIZE      Size of a chunk for chunk-based HTTP
                                downloading, e.g. 10485760 or 10M (default
                                is disabled). May be useful for bypassing
                                bandwidth throttling imposed by a webserver
                                (experimental)
    --playlist-random           Download playlist videos in random order
    --lazy-playlist             Process entries in the playlist as they are
                                received. This disables n_entries,
                                --playlist-random and --playlist-reverse
    --no-lazy-playlist          Process videos in the playlist only after
                                the entire playlist is parsed (default)
    --xattr-set-filesize        Set file xattribute ytdl.filesize with
                                expected file size
    --hls-use-mpegts            Use the mpegts container for HLS videos;
                                allowing some players to play the video
                                while downloading, and reducing the chance
                                of file corruption if download is
                                interrupted. This is enabled by default for
                                live streams
    --no-hls-use-mpegts         Do not use the mpegts container for HLS
                                videos. This is default when not downloading
                                live streams
    --download-sections REGEX   Download only chapters that match the
                                regular expression. A "*" prefix denotes
                                time-range instead of chapter. Negative
                                timestamps are calculated from the end.
                                "*from-url" can be used to download between
                                the "start_time" and "end_time" extracted
                                from the URL. Needs ffmpeg. This option can
                                be used multiple times to download multiple
                                sections, e.g. --download-sections
                                "*10:15-inf" --download-sections "intro"
    --downloader [PROTO:]NAME   Name or path of the external downloader to
                                use (optionally) prefixed by the protocols
                                (http, ftp, m3u8, dash, rstp, rtmp, mms) to
                                use it for. Currently supports native,
                                aria2c, avconv, axel, curl, ffmpeg, httpie,
                                wget. You can use this option multiple times
                                to set different downloaders for different
                                protocols. E.g. --downloader aria2c
                                --downloader "dash,m3u8:native" will use
                                aria2c for http/ftp downloads, and the
                                native downloader for dash/m3u8 downloads
                                (Alias: --external-downloader)
    --downloader-args NAME:ARGS Give these arguments to the external
                                downloader. Specify the downloader name and
                                the arguments separated by a colon ":". For
                                ffmpeg, arguments can be passed to different
                                positions using the same syntax as
                                --postprocessor-args. You can use this
                                option multiple times to give different
                                arguments to different downloaders (Alias:
                                --external-downloader-args)
```

Параметры файловой системы:

```
    -a, --batch-file FILE       File containing URLs to download ("-" for
                                stdin), one URL per line. Lines starting
                                with "#", ";" or "]" are considered as
                                comments and ignored
    --no-batch-file             Do not read URLs from batch file (default)
    -P, --paths [TYPES:]PATH    The paths where the files should be
                                downloaded. Specify the type of file and the
                                path separated by a colon ":". All the same
                                TYPES as --output are supported.
                                Additionally, you can also provide "home"
                                (default) and "temp" paths. All intermediary
                                files are first downloaded to the temp path
                                and then the final files are moved over to
                                the home path after download is finished.
                                This option is ignored if --output is an
                                absolute path
    -o, --output [TYPES:]TEMPLATE Output filename template; see "OUTPUT
                                TEMPLATE" for details
```

```
    --output-na-placeholder TEXT    Placeholder for unavailable fields in
                                    --output (default: "NA")
    --restrict-filenames            Restrict filenames to only ASCII characters,
                                    and avoid "&" and spaces in filenames
    --no-restrict-filenames         Allow Unicode characters, "&" and spaces in
                                    filenames (default)
    --windows-filenames             Force filenames to be Windows-compatible
    --no-windows-filenames          Make filenames Windows-compatible only if
                                    using Windows (default)
    --trim-filenames LENGTH         Limit the filename length (excluding
                                    extension) to the specified number of
                                    characters
    -w, --no-overwrites             Do not overwrite any files
    --force-overwrites              Overwrite all video and metadata files. This
                                    option includes --no-continue
    --no-force-overwrites           Do not overwrite the video, but overwrite
                                    related files (default)
    -c, --continue                  Resume partially downloaded files/fragments
                                    (default)
    --no-continue                   Do not resume partially downloaded
                                    fragments. If the file is not fragmented,
                                    restart download of the entire file
    --part                          Use .part files instead of writing directly
                                    into output file (default)
    --no-part                       Do not use .part files - write directly into
                                    output file
    --mtime                         Use the Last-modified header to set the file
                                    modification time (default)
    --no-mtime                      Do not use the Last-modified header to set
                                    the file modification time
    --write-description             Write video description to a .description file
    --no-write-description          Do not write video description (default)
    --write-info-json               Write video metadata to a .info.json file
                                    (this may contain personal information)
    --no-write-info-json            Do not write video metadata (default)
    --write-playlist-metafiles      Write playlist metadata in addition to the
                                    video metadata when using --write-info-json,
                                    --write-description etc. (default)
    --no-write-playlist-metafiles   Do not write playlist metadata when using
                                    --write-info-json, --write-description etc.
    --clean-info-json               Remove some internal metadata such as
                                    filenames from the infojson (default)
    --no-clean-info-json            Write all fields to the infojson
    --write-comments                Retrieve video comments to be placed in the
                                    infojson. The comments are fetched even
                                    without this option if the extraction is
                                    known to be quick (Alias: --get-comments)
    --no-write-comments             Do not retrieve video comments unless the
                                    extraction is known to be quick (Alias:
                                    --no-get-comments)
    --load-info-json FILE           JSON file containing the video information
                                    (created with the "--write-info-json" option)
    --cookies FILE                  Netscape formatted file to read cookies from
                                    and dump cookie jar in
    --no-cookies                    Do not read/dump cookies from/to file
                                    (default)
    --cookies-from-browser BROWSER[+KEYRING][:PROFILE][::CONTAINER]
                                    The name of the browser to load cookies
                                    from. Currently supported browsers are:
                                    brave, chrome, chromium, edge, firefox,
                                    opera, safari, vivaldi, whale. Optionally,
                                    the KEYRING used for decrypting Chromium
                                    cookies on Linux, the name/path of the
                                    PROFILE to load cookies from, and the
                                    CONTAINER name (if Firefox) ("none" for no
                                    container) can be given with their
                                    respective separators. By default, all
                                    containers of the most recently accessed
                                    profile are used. Currently supported
                                    keyrings are: basictext, gnomekeyring,
                                    kwallet, kwallet5, kwallet6
    --no-cookies-from-browser       Do not load cookies from browser (default)
    --cache-dir DIR                 Location in the filesystem where yt-dlp can
                                    store some downloaded information (such as
                                    client ids and signatures) permanently. By
                                    default ${XDG_CACHE_HOME}/yt-dlp
    --no-cache-dir                  Disable filesystem caching
    --rm-cache-dir                  Delete all filesystem cache files
```

Параметры миниатюр:

```
--write-thumbnail               Write thumbnail image to disk
--no-write-thumbnail            Do not write thumbnail image to disk (default)
--write-all-thumbnails          Write all thumbnail image formats to disk
--list-thumbnails               List available thumbnails of each video.
                                Simulate unless --no-simulate is used
```

Параметры сочетания клавиш Интернета:

```
--write-link                    Write an internet shortcut file, depending
                                on the current platform (.url, .webloc or
                                .desktop). The URL may be cached by the OS
--write-url-link                Write a .url Windows internet shortcut. The
                                OS caches the URL based on the file path
--write-webloc-link             Write a .webloc macOS internet shortcut
--write-desktop-link            Write a .desktop Linux internet shortcut
```

Параметры детализации и моделирования:

```
-q, --quiet                     Activate quiet mode. If used with --verbose,
                                print the log to stderr
--no-quiet                      Deactivate quiet mode. (Default)
--no-warnings                   Ignore warnings
-s, --simulate                  Do not download the video and do not write
                                anything to disk
--no-simulate                   Download the video even if printing/listing
                                options are used
--ignore-no-formats-error       Ignore "No video formats" error. Useful for
                                extracting metadata even if the videos are
                                not actually available for download
                                (experimental)
--no-ignore-no-formats-error    Throw error when no downloadable video
                                formats are found (default)
--skip-download                 Do not download the video but write all
                                related files (Alias: --no-download)
-O, --print [WHEN:]TEMPLATE     Field name or output template to print to
                                screen, optionally prefixed with when to
                                print it, separated by a ":". Supported
                                values of "WHEN" are the same as that of
                                --use-postprocessor (default: video).
                                Implies --quiet. Implies --simulate unless
                                --no-simulate or later stages of WHEN are
                                used. This option can be used multiple times
--print-to-file [WHEN:]TEMPLATE FILE
                                Append given template to the file. The
                                values of WHEN and TEMPLATE are same as that
                                of --print. FILE uses the same syntax as the
                                output template. This option can be used
```

ПРОЧТИ МЕНЯ Нелицензированная лицензия

```
                                video. Simulate unless --no-simulate is
                                used. See "OUTPUT TEMPLATE" for a
                                description of available keys
-J, --dump-single-json          Quiet, but print JSON information for each
                                url or infojson passed. Simulate unless
                                --no-simulate is used. If the URL refers to
                                a playlist, the whole playlist information
                                is dumped in a single line
--force-write-archive           Force download archive entries to be written
                                as far as no errors occur, even if -s or
                                another simulation option is used (Alias:
                                --force-download-archive)
--newline                       Output progress bar as new lines
--no-progress                   Do not print progress bar
--progress                      Show progress bar, even if in quiet mode
--console-title                 Display progress in console titlebar
--progress-template [TYPES:]TEMPLATE
                                Template for progress outputs, optionally
                                prefixed with one of "download:" (default),
                                "download-title:" (the console title),
```

```
                                  "postprocess:",  or "postprocess-title:".
                                  The video's fields are accessible under the
                                  "info" key and the progress attributes are
                                  accessible under "progress" key. E.g.
                                  --console-title --progress-template
                                  "download-title:%(info.id)s-%(progress.eta)s"
--progress-delta SECONDS          Time between progress output (default: 0)
-v, --verbose                     Print various debugging information
--dump-pages                      Print downloaded pages encoded using base64
                                  to debug problems (very verbose)
--write-pages                     Write downloaded intermediary pages to files
                                  in the current directory to debug problems
--print-traffic                   Display sent and read HTTP traffic
```

Обходные пути:

```
--encoding ENCODING               Force the specified encoding (experimental)
--legacy-server-connect           Explicitly allow HTTPS connection to servers
                                  that do not support RFC 5746 secure
                                  renegotiation
--no-check-certificates           Suppress HTTPS certificate validation
--prefer-insecure                 Use an unencrypted connection to retrieve
                                  information about the video (Currently
                                  supported only for YouTube)
--add-headers FIELD:VALUE         Specify a custom HTTP header and its value,
                                  separated by a colon ":". You can use this
                                  option multiple times
--bidi-workaround                 Work around terminals that lack
                                  bidirectional text support. Requires bidiv
                                  or fribidi executable in PATH
--sleep-requests SECONDS          Number of seconds to sleep between requests
                                  during data extraction
--sleep-interval SECONDS          Number of seconds to sleep before each
                                  download. This is the minimum time to sleep
                                  when used along with --max-sleep-interval
                                  (Alias: --min-sleep-interval)
--max-sleep-interval SECONDS      Maximum number of seconds to sleep. Can only
                                  be used along with --min-sleep-interval
--sleep-subtitles SECONDS         Number of seconds to sleep before each
                                  subtitle download
```

Параметры формата видео:

```
-f, --format FORMAT               Video format code, see "FORMAT SELECTION"
                                  for more details
-S, --format-sort SORTORDER       Sort the formats by the fields given, see
                                  "Sorting Formats" for more details
--format-sort-force               Force user specified sort order to have
                                  precedence over all fields, see "Sorting
                                  Formats" for more details (Alias: --S-force)
--no-format-sort-force            Some fields have precedence over the user
                                  specified sort order (default)
--video-multistreams              Allow multiple video streams to be merged
                                  into a single file
--no-video-multistreams           Only one video stream is downloaded for each
                                  output file (default)
--audio-multistreams              Allow multiple audio streams to be merged
                                  into a single file
--no-audio-multistreams           Only one audio stream is downloaded for each
                                  output file (default)
--prefer-free-formats             Prefer video formats with free containers
                                  over non-free ones of same quality. Use with
                                  "-S ext" to strictly prefer free containers
                                  irrespective of quality
--no-prefer-free-formats          Don't give any special preference to free
                                  containers (default)
--check-formats                   Make sure formats are selected only from
                                  those that are actually downloadable
--check-all-formats               Check all formats for whether they are
                                  actually downloadable
--no-check-formats                Do not check that the formats are actually
                                  downloadable
-F, --list-formats                List available formats of each video.
                                  Simulate unless --no-simulate is used
```

```
--merge-output-format FORMAT    Containers that may be used when merging
                                formats, separated by "/", e.g. "mp4/mkv".
                                Ignored if no merge is required. (currently
                                supported: avi, flv, mkv, mov, mp4, webm)
```

Параметры субтитров:

```
--write-subs                    Write subtitle file
--no-write-subs                 Do not write subtitle file (default)
--write-auto-subs               Write automatically generated subtitle file
                                (Alias: --write-automatic-subs)
--no-write-auto-subs            Do not write auto-generated subtitles
                                (default) (Alias: --no-write-automatic-subs)
--list-subs                     List available subtitles of each video.
                                Simulate unless --no-simulate is used
--sub-format FORMAT             Subtitle format; accepts formats preference,
                                e.g. "srt" or "ass/srt/best"
--sub-langs LANGS               Languages of the subtitles to download (can
                                be regex) or "all" separated by commas, e.g.
                                --sub-langs "en.*,ja". You can prefix the
                                language code with a "-" to exclude it from
                                the requested languages, e.g. --sub-langs
                                all,-live_chat. Use --list-subs for a list
                                of available language tags
```

Варианты аутентификации:

```
-u, --username USERNAME         Login with this account ID
-p, --password PASSWORD         Account password. If this option is left
                                out, yt-dlp will ask interactively
-2, --twofactor TWOFACTOR       Two-factor authentication code
-n, --netrc                     Use .netrc authentication data
--netrc-location PATH           Location of .netrc authentication data;
                                either the path or its containing directory.
                                Defaults to ~/.netrc
--netrc-cmd NETRC_CMD           Command to execute to get the credentials
                                for an extractor.
--video-password PASSWORD       Video-specific password
--ap-mso MSO                    Adobe Pass multiple-system operator (TV
                                provider) identifier, use --ap-list-mso for
                                a list of available MSOs
--ap-username USERNAME          Multiple-system operator account login
--ap-password PASSWORD          Multiple-system operator account password.
                                If this option is left out, yt-dlp will ask
                                interactively
--ap-list-mso                   List all supported multiple-system operators
--client-certificate CERTFILE   Path to client certificate file in PEM
                                format. May include the private key
--client-certificate-key KEYFILE
                                Path to private key file for client
                                certificate
--client-certificate-password PASSWORD
                                Password for client certificate private key,
                                if encrypted. If not provided, and the key
                                is encrypted, yt-dlp will ask interactively
```

Варианты постобработки:

```
-x, --extract-audio             Convert video files to audio-only files
                                (requires ffmpeg and ffprobe)
--audio-format FORMAT           Format to convert the audio to when -x is
                                used. (currently supported: best (default),
                                aac, alac, flac, m4a, mp3, opus, vorbis,
                                wav). You can specify multiple rules using
                                similar syntax as --remux-video
--audio-quality QUALITY         Specify ffmpeg audio quality to use when
                                converting the audio with -x. Insert a value
                                between 0 (best) and 10 (worst) for VBR or a
                                specific bitrate like 128K (default 5)
--remux-video FORMAT            Remux the video into another container if
                                necessary (currently supported: avi, flv,
```

```
                                gif, mkv, mov, mp4, webm, aac, aiff, alac,
                                flac, m4a, mka, mp3, ogg, opus, vorbis,
                                wav). If target container does not support
                                the video/audio codec, remuxing will fail.
                                You can specify multiple rules; e.g.
                                "aac>m4a/mov>mp4/mkv" will remux aac to m4a,
                                mov to mp4 and anything else to mkv
    --recode-video FORMAT       Re-encode the video into another format if
                                necessary. The syntax and supported formats
                                are the same as --remux-video
    --postprocessor-args NAME:ARGS  Give these arguments to the postprocessors.
                                Specify the postprocessor/executable name
                                and the arguments separated by a colon ":"
                                to give the argument to the specified
                                postprocessor/executable. Supported PP are:
                                Merger, ModifyChapters, SplitChapters,
                                ExtractAudio, VideoRemuxer, VideoConvertor,
                                Metadata, EmbedSubtitle, EmbedThumbnail,
                                SubtitlesConvertor, ThumbnailsConvertor,
                                FixupStretched, FixupM4a, FixupM3u8,
                                FixupTimestamp and FixupDuration. The
                                supported executables are: AtomicParsley,
                                FFmpeg and FFprobe. You can also specify
                                "PP+EXE:ARGS" to give the arguments to the
                                specified executable only when being used by
                                the specified postprocessor. Additionally,
                                for ffmpeg/ffprobe, "_i"/"_o" can be
                                appended to the prefix optionally followed
                                by a number to pass the argument before the
                                specified input/output file, e.g. --ppa
                                "Merger+ffmpeg_i1:-v quiet". You can use
                                this option multiple times to give different
                                arguments to different postprocessors.
                                (Alias: --ppa)
    -k, --keep-video            Keep the intermediate video file on disk
                                after post-processing
    --no-keep-video             Delete the intermediate video file after
                                post-processing (default)
    --post-overwrites           Overwrite post-processed files (default)
    --no-post-overwrites        Do not overwrite post-processed files
    --embed-subs                Embed subtitles in the video (only for mp4,
                                webm and mkv videos)
    --no-embed-subs             Do not embed subtitles (default)
    --embed-thumbnail           Embed thumbnail in the video as cover art
    --no-embed-thumbnail        Do not embed thumbnail (default)
    --embed-metadata            Embed metadata to the video file. Also
                                embeds chapters/infojson if present unless
                                --no-embed-chapters/--no-embed-info-json are
                                used (Alias: --add-metadata)
    --no-embed-metadata         Do not add metadata to file (default)
                                (Alias: --no-add-metadata)
    --embed-chapters            Add chapter markers to the video file
                                (Alias: --add-chapters)
    --no-embed-chapters         Do not add chapter markers (default) (Alias:
                                --no-add-chapters)
    --embed-info-json           Embed the infojson as an attachment to
                                mkv/mka video files
    --no-embed-info-json        Do not embed the infojson as an attachment
                                to the video file
    --parse-metadata [WHEN:]FROM:TO
                                Parse additional metadata like title/artist
                                from other fields; see "MODIFYING METADATA"
                                for details. Supported values of "WHEN" are
                                the same as that of --use-postprocessor
                                (default: pre_process)
    --replace-in-metadata [WHEN:]FIELDS REGEX REPLACE
                                Replace text in a metadata field using the
                                given regex. This option can be used
                                multiple times. Supported values of "WHEN"
                                are the same as that of --use-postprocessor
                                (default: pre_process)
    --xattrs                    Write metadata to the video file's xattrs
                                (using dublin core and xdg standards)
    --concat-playlist POLICY    Concatenate videos in a playlist. One of
                                "never", "always", or "multi_video"
                                (default; only when the videos form a single
                                show). All the video files must have same
                                codecs and number of streams to be
                                concatable. The "pl_video:" prefix can be
```

```
                                used with "--paths" and "--output" to set
                                the output filename for the concatenated
                                files. See "OUTPUT TEMPLATE" for details
    --fixup POLICY              Automatically correct known faults of the
                                file. One of never (do nothing), warn (only
                                emit a warning), detect_or_warn (the
                                default; fix file if we can, warn
                                otherwise), force (try fixing even if file
                                already exists)
    --ffmpeg-location PATH      Location of the ffmpeg binary; either the
                                path to the binary or its containing directory
    --exec [WHEN:]CMD           Execute a command, optionally prefixed with
                                when to execute it, separated by a ":".
                                Supported values of "WHEN" are the same as
                                that of --use-postprocessor (default:
                                after_move). Same syntax as the output
                                template can be used to pass any field as
                                arguments to the command. If no fields are
                                passed, %(filepath,_filename|)q is appended
                                to the end of the command. This option can
                                be used multiple times
    --no-exec                   Remove any previously defined --exec
    --convert-subs FORMAT       Convert the subtitles to another format
                                (currently supported: ass, lrc, srt, vtt)
                                (Alias: --convert-subtitles)
    --convert-thumbnails FORMAT Convert the thumbnails to another format
                                (currently supported: jpg, png, webp). You
                                can specify multiple rules using similar
                                syntax as --remux-video
    --split-chapters            Split video into multiple files based on
                                internal chapters. The "chapter:" prefix can
                                be used with "--paths" and "--output" to set
                                the output filename for the split files. See
                                "OUTPUT TEMPLATE" for details
    --no-split-chapters         Do not split video based on chapters (default)
    --remove-chapters REGEX     Remove chapters whose title matches the
                                given regular expression. The syntax is the
                                same as --download-sections. This option can
                                be used multiple times
    --no-remove-chapters        Do not remove any chapters from the file
                                (default)
    --force-keyframes-at-cuts   Force keyframes at cuts when
                                downloading/splitting/removing sections.
                                This is slow due to needing a re-encode, but
                                the resulting video may have fewer artifacts
                                around the cuts
    --no-force-keyframes-at-cuts Do not force keyframes around the chapters
                                when cutting/splitting (default)
    --use-postprocessor NAME[:ARGS]
                                The (case sensitive) name of plugin
                                postprocessors to be enabled, and
                                (optionally) arguments to be passed to it,
                                separated by a colon ":". ARGS are a
                                semicolon ";" delimited list of NAME=VALUE.
                                The "when" argument determines when the
                                postprocessor is invoked. It can be one of
                                "pre_process" (after video extraction),
                                "after_filter" (after video passes filter),
                                "video" (after --format; before
                                --print/--output), "before_dl" (before each
                                video download), "post_process" (after each
                                video download; default), "after_move"
                                (after moving video file to its final
                                locations), "after_video" (after downloading
                                and processing all formats of a video), or
                                "playlist" (at end of playlist). This option
                                can be used multiple times to add different
                                postprocessors
```

Параметры SponsorBlock:

Создавайте записи глав или удаляйте различные сегменты (спонсоры, вступления и т. д.) из загруженных видео YouTube с помощью API SponsorBlock.

```
    --sponsorblock-mark CATS    SponsorBlock categories to create chapters
                                for, separated by commas. Available
```

```
                                categories are sponsor, intro, outro,
                                selfpromo, preview, filler, interaction,
                                music_offtopic, poi_highlight, chapter, all
                                and default (=all). You can prefix the
                                category with a "-" to exclude it. See [1]
                                for description of the categories. E.g.
                                --sponsorblock-mark all,-preview
                                [1] https://wiki.sponsor.ajay.app/w/Segment_Categories
    --sponsorblock-remove CATS  SponsorBlock categories to be removed from
                                the video file, separated by commas. If a
                                category is present in both mark and remove,
                                remove takes precedence. The syntax and
                                available categories are the same as for
                                --sponsorblock-mark except that "default"
                                refers to "all,-filler" and poi_highlight,
                                chapter are not available
    --sponsorblock-chapter-title TEMPLATE
                                An output template for the title of the
                                SponsorBlock chapters created by
                                --sponsorblock-mark. The only available
                                fields are start_time, end_time, category,
                                categories, name, category_names. Defaults
                                to "[SponsorBlock]: %(category_names)l"
    --no-sponsorblock           Disable both --sponsorblock-mark and
                                --sponsorblock-remove
    --sponsorblock-api URL      SponsorBlock API location, defaults to
                                https://sponsor.ajay.app
```

Параметры экстрактора:

```
    --extractor-retries RETRIES     Number of retries for known extractor errors
                                    (default is 3), or "infinite"
    --allow-dynamic-mpd             Process dynamic DASH manifests (default)
                                    (Alias: --no-ignore-dynamic-mpd)
    --ignore-dynamic-mpd            Do not process dynamic DASH manifests
                                    (Alias: --no-allow-dynamic-mpd)
    --hls-split-discontinuity       Split HLS playlists to different formats at
                                    discontinuities such as ad breaks
    --no-hls-split-discontinuity    Do not split HLS playlists to different
                                    formats at discontinuities such as ad breaks
                                    (default)
    --extractor-args IE_KEY:ARGS    Pass ARGS arguments to the IE_KEY extractor.
                                    See "EXTRACTOR ARGUMENTS" for details. You
                                    can use this option multiple times to give
                                    arguments for different extractors
```

КОНФИГУРАЦИЯ

Вы можете настроить yt-dlp, поместив любой поддерживаемый параметр командной строки в файл конфигурации. Конфигурация загружается из следующих мест:

1. Основная конфигурация :
    - Файл, предоставленный `--config-location`
2. Портативная конфигурация : (рекомендуется для портативных установок)
    - Если используется двоичный файл, `yt-dlp.conf` в том же каталоге, что и двоичный файл
    - Если запуск из исходного кода, `yt-dlp.conf` в родительском каталоге `yt_dlp`
3. Домашняя конфигурация :
    - `yt-dlp.conf` в домашнем пути, данном `-P`
    - Если `-P` не указано, поиск выполняется в текущем каталоге.
4. Конфигурация пользователя :
    - `${XDG_CONFIG_HOME}/yt-dlp.conf`
    - `${XDG_CONFIG_HOME}/yt-dlp/config` (рекомендуется для Linux/macOS)
    - `${XDG_CONFIG_HOME}/yt-dlp/config.txt`
    - `${APPDATA}/yt-dlp.conf`

- ${APPDATA}/yt-dlp/config (рекомендуется для Windows)
- ${APPDATA}/yt-dlp/config.txt
- ~/yt-dlp.conf
- ~/yt-dlp.conf.txt
- ~/.yt-dlp/config
- ~/.yt-dlp/config.txt

См. также: Заметки о переменных окружения

5. Конфигурация системы :
   - /etc/yt-dlp.conf
   - /etc/yt-dlp/config
   - /etc/yt-dlp/config.txt

Например, при использовании следующего файла конфигурации yt-dlp всегда будет извлекать аудио, не копировать mtime, использовать прокси-сервер и сохранять все видео в YouTube каталоге в вашем домашнем каталоге:

```
# Lines starting with # are comments

# Always extract audio
-x

# Do not copy the mtime
--no-mtime

# Use this proxy
--proxy 127.0.0.1:3128

# Save all videos under YouTube directory in your home directory
-o ~/YouTube/%(title)s.%(ext)s
```

Примечание : Параметры в файле конфигурации — это те же параметры, которые используются в обычных вызовах командной строки; таким образом, после or не должно быть пробелов , например, or , но не or . При необходимости их также следует заключать в кавычки, как если бы это была оболочка UNIX. - -- -o --proxy - o -- proxy

Вы можете использовать --ignore-config , если хотите отключить все файлы конфигурации для конкретного запуска yt-dlp. Если , --ignore-config найдено внутри любого файла конфигурации, никакая дополнительная конфигурация не будет загружена. Например, наличие опции в переносимом файле конфигурации предотвращает загрузку домашней, пользовательской и системной конфигураций. Кроме того, (для обратной совместимости) если , --ignore-config найдено внутри файла конфигурации системы, конфигурация пользователя не загружается.

Кодировка файла конфигурации

Файлы конфигурации декодируются в соответствии с кодировкой UTF BOM, если она присутствует, и в соответствии с кодировкой локали системы в противном случае.

Если вы хотите, чтобы ваш файл был декодирован по-другому, добавьте # coding: ENCODING в начало файла (например # coding: shift-jis). Перед этим не должно быть никаких символов, даже пробелов или BOM.

Аутентификация с помощью netrc

Вы также можете настроить автоматическое хранение учетных данных для экстракторов, которые поддерживают аутентификацию (предоставляя логин и пароль с помощью --username и --password), чтобы не передавать учетные данные в качестве аргументов командной строки при каждом выполнении yt-dlp и предотвратить отслеживание текстовых паролей в истории команд оболочки. Вы можете добиться этого, используя .netrc файл для каждого экстрактора. Для этого вам нужно будет создать файл .netrc и --netrc-location ограничить разрешения на чтение/запись только вами:

```
touch ${HOME}/.netrc
chmod a-rwx,u+rw ${HOME}/.netrc
```

После этого вы можете добавить учетные данные для экстрактора в следующем формате, где *extractor* — это имя экстрактора в нижнем регистре:

```
machine <extractor> login <username> password <password>
```

Например

```
machine youtube login myaccount@gmail.com password my_youtube_password
machine twitch login my_twitch_account_name password my_twitch_password
```

Для активации аутентификации с помощью `.netrc` файла необходимо перейти `--netrc` к yt-dlp или поместить его в файл конфигурации .

Расположение файла .netrc по умолчанию: `~` (см. ниже).

В качестве альтернативы использованию `.netrc` файла, недостаток которого заключается в хранении паролей в текстовом файле, вы можете настроить пользовательскую команду оболочки для предоставления учетных данных для экстрактора. Это делается путем предоставления параметра `--netrc-cmd` , он должен выводить учетные данные в формате netrc и возвращать `0` в случае успеха, другие значения будут рассматриваться как ошибка. `{}` в команде будет заменено именем экстрактора, чтобы можно было выбрать учетные данные для нужного экстрактора.

Например, чтобы использовать зашифрованный `.netrc` файл, сохраненный как `.authinfo.gpg`

```
yt-dlp --netrc-cmd 'gpg --decrypt ~/.authinfo.gpg' https://www.youtube.com/watch?v=BaW_jenozKc
```

Заметки о переменных среды

- Переменные среды обычно указываются как `${VARIABLE}` / `$VARIABLE` в UNIX и `%VARIABLE%` Windows; но `${VARIABLE}` в этой документации они всегда отображаются как
- yt-dlp также позволяет использовать переменные в стиле UNIX в Windows для параметров, подобных путям; например `--output` , `--config-location`
- Если не установлено, по `${XDG_CONFIG_HOME}` умолчанию используются `~/.config` и `${XDG_CACHE_HOME}` `~/.cache`
- В Windows `~` указывает на , `${HOME}` если присутствует; или, `${USERPROFILE}` или в `${HOMEDRIVE}${HOMEPATH}` противном случае
- В Windows `${USERPROFILE}` обычно указывает на `C:\Users\<user name>` и `${APPDATA}` на `${USERPROFILE}\AppData\Roaming`

ВЫХОДНОЙ ШАБЛОН

Параметр `-o` используется для указания шаблона для имен выходных файлов, а `-P` параметр используется для указания пути, по которому следует сохранять каждый тип файла.

tl;dr: перенаправьте меня к примерам .

Простейший способ использования `-o` — не задавать никаких аргументов шаблона при загрузке отдельного файла, как в `yt-dlp -o funny_video.flv "https://some/video"` (жесткое кодирование расширения файла, подобное этому, *не* рекомендуется и может привести к сбоям в постобработке).

Однако он может также содержать специальные последовательности, которые будут заменены при загрузке каждого видео. Специальные последовательности могут быть отформатированы в соответствии с операциями форматирования строк Python , например, `%(NAME)s` или `%(NAME)05d` . Для ясности, это символ процента, за которым следует имя в скобках, за которым следуют операции форматирования.

Сами имена полей (часть внутри скобок) также могут иметь особое форматирование:

1. Обход объекта : словари и списки, доступные в метаданных, можно обходить с помощью `.` разделителя-точки; например `%(tags.0)s` , `%(subtitles.en.-1.ext)s` . Вы можете выполнить срез Python с помощью двоеточия `:` ; например `%(id.3:7)s` , `%(id.6:2:-1)s` , `%(formats.:.format_id)s` . Фигурные скобки `{}` можно использовать для создания словарей только с определенными ключами; например `%(formats.:.{format_id,height})#j` , . Пустое имя поля `%()s` относится ко всему infodict; например `%(.{id,title})s` , . Обратите внимание, что все поля, которые становятся доступными с помощью этого метода, не перечислены ниже. Используйте `-j` для просмотра таких полей

2. Арифметика : Простые арифметические действия можно выполнять с числовыми полями, используя `+` , `-` и `*` . Например `%(playlist_index+10)03d` , `%(n_entries+1-playlist_index)d`

3. Форматирование даты/времени : поля даты/времени можно форматировать в соответствии с форматированием strftime , указав его отдельно от имени поля с помощью `>` . Например `%(duration>%H-%M-%S)s` , `%(upload_date>%Y-%m-%d)s` , , `%(epoch-3600>%H-%M-%S)s`

4. Альтернативы : Альтернативные поля могут быть указаны через точку `,` . Например: `%(release_date>%Y,upload_date>%Y|Unknown)s`

5. Замена : значение замены может быть указано с помощью `&` разделителя в соответствии с `str.format` мини-языком . Если поле *не* пустое, это значение замены будет использоваться вместо фактического содержимого поля. Это делается после рассмотрения альтернативных полей; таким образом, замена используется, если *любое* из альтернативных полей *не* пустое. Например `%(chapters&has chapters|no chapters)s` , `%(title&TITLE={:>20}|NO TITLE)s`

6. По умолчанию : можно указать буквальное значение по умолчанию, если поле пустое, с помощью `|` разделителя. Это переопределяет `--output-na-placeholder` . Например `%(uploader|Unknown)s`

7. Дополнительные преобразования : в дополнение к обычным типам формата `diouxXeEfFgGcrs` yt-dlp дополнительно поддерживает преобразование в `B` = байты , `j` = json (флаг `#` для красивой печати, `+` для Unicode), `h` = экранирование HTML, `=` список `l` , разделенный запятыми (флаг для разделения новой строкой), `=` строку , заключенную в кавычки для терминала (флаг для разделения списка на разные аргументы), `=` добавление десятичных суффиксов (например, 10M) (флаг для использования 1024 в качестве множителя) и `=` sanitize как имя файла (флаг для ограничения) `# \n q # D # S #`

8. Нормализация Unicode : Тип формата `U` может использоваться для нормализации Unicode NFC . Флаг альтернативной формы (`#`) изменяет нормализацию на NFD, а флаг преобразования `+` может использоваться для нормализации эквивалентности совместимости NFKC/NFKD. Например, `%(title)+.100U` NFKC

Подводя итог, общий синтаксис поля следующий:

```
%(name[.keys][addition][>strf][,alternate][&replacement][|default])[flags][width][.precision][length]type
```

Кроме того, вы можете задать различные выходные шаблоны для различных файлов метаданных отдельно от общего выходного шаблона, указав тип файла, за которым следует шаблон, разделенный двоеточием `:` . Поддерживаются различные типы файлов `subtitle` : `thumbnail` , , `description` , `annotation` (устарело), `infojson` , `link` , `pl_thumbnail` , `pl_description` , `pl_infojson` , `chapter` , `pl_video` . Например, `-o "%(title)s.%(ext)s" -o "thumbnail:%(title)s\%(title)s.%(ext)s"` миниатюры будут помещены в папку с тем же именем, что и у видео. Если какой-либо из шаблонов пуст, этот тип файла не будет записан. Например, `--write-thumbnail -o "thumbnail:"` миниатюры будут записаны только для плейлистов, но не для видео.

Примечание : Из-за постобработки (т.е. слияния и т.п.) фактическое имя выходного файла может отличаться. Используйте `--print after_move:filepath` для получения имени после завершения всей постобработки.

Доступные поля:

- `id` (строка): Идентификатор видео
- `title` (строка): Название видео
- `fulltitle` (строка): Название видео, игнорирующее метку времени в реальном времени и общее название
- `ext` (строка): Расширение имени видеофайла
- `alt_title` (строка): Вторичное название видео.
- `description` (строка): Описание видео
- `display_id` (строка): Альтернативный идентификатор видео.
- `uploader` (строка): Полное имя загрузчика видео
- `uploader_id` (строка): Псевдоним или идентификатор пользователя, загрузившего видео
- `uploader_url` (строка): URL-адрес профиля загрузчика видео
- `license` (строка): Название лицензии, по которой распространяется видео
- `creators` (список): Создатели видео
- `creator` (строка): Создатели видео; через запятую
- `timestamp` (числовое): временная метка UNIX момента, когда видео стало доступно
- `upload_date` (строка): Дата загрузки видео в формате UTC (ГГГГММДД)
- `release_timestamp` (числовое): временная метка UNIX момента выхода видео
- `release_date` (строка): Дата (ГГГГММДД) выхода видео по UTC
- `release_year` (числовой): Год (ГГГГ), когда было выпущено видео или альбом
- `modified_timestamp` (числовое): временная метка UNIX момента последнего изменения видео.
- `modified_date` (строка): Дата (ГГГГММДД) последнего изменения видео в формате UTC
- `channel` (строка): Полное название канала, на котором загружено видео
- `channel_id` (строка): Идентификатор канала
- `channel_url` (строка): URL канала
- `channel_follower_count` (числовое): Количество подписчиков канала
- `channel_is_verified` (логическое значение): проверен ли канал на платформе
- `location` (строка): Физическое место, где было снято видео
- `duration` (числовое): Продолжительность видео в секундах.
- `duration_string` (строка): Продолжительность видео (ЧЧ:мм:сс)

- `view_count` (числовое значение): Сколько пользователей посмотрели видео на платформе
- `concurrent_view_count` (числовое значение): Сколько пользователей в данный момент смотрят видео на платформе.
- `like_count` (числовое): Количество положительных оценок видео
- `dislike_count` (числовое): Количество отрицательных оценок видео
- `repost_count` (числовое): Количество репостов видео
- `average_rating` (числовой): Средняя оценка, данная пользователями, используемая шкала зависит от веб-страницы.
- `comment_count` (числовое): Количество комментариев к видео (для некоторых экстракторов комментарии загружаются только в конце, поэтому это поле не может быть использовано)
- `age_limit` (числовое): Возрастное ограничение для видео (лет)
- `live_status` (строка): Одно из "not_live", "is_live", "is_upcoming", "was_live", "post_live" (был в прямом эфире, но VOD еще не обработан)
- `is_live` (булево): Является ли это видео прямой трансляцией или видео фиксированной длины
- `was_live` (булево): Было ли это видео изначально прямой трансляцией
- `playable_in_embed` (строка): Разрешено ли воспроизводить это видео во встроенных проигрывателях на других сайтах
- `availability` (строка): Является ли видео «частным», «только для премиум-пользователей», «только для подписчиков», «требует_авторизации», «не в списке» или «публичным»
- `media_type` (строка): Тип носителя, классифицированный сайтом, например, «эпизод», «клип», «трейлер»
- `start_time` (числовое): Время в секундах, когда должно начаться воспроизведение, как указано в URL
- `end_time` (числовое): Время в секундах, когда воспроизведение должно закончиться, как указано в URL
- `extractor` (строка): Имя экстрактора
- `extractor_key` (строка): Имя ключа экстрактора
- `epoch` (числовое): Эпоха Unix, когда было завершено извлечение информации.
- `autonumber` (числовой): Число, которое будет увеличиваться с каждой загрузкой, начиная с `--autonumber-start`, дополненное начальными нулями до 5 цифр
- `video_autonumber` (числовое): Число, которое будет увеличиваться с каждым видео
- `n_entries` (числовое): Общее количество извлеченных элементов в плейлисте
- `playlist_id` (строка): Идентификатор плейлиста, содержащего видео
- `playlist_title` (строка): Имя плейлиста, содержащего видео
- `playlist` (строка): `playlist_title` если доступно или иначе `playlist_id`
- `playlist_count` (числовое): Общее количество элементов в плейлисте. Может быть неизвестно, если весь плейлист не извлечен
- `playlist_index` (числовой): Индекс видео в плейлисте, дополненный ведущими нулями в соответствии с конечным индексом.
- `playlist_autonumber` (числовое): Позиция видео в очереди загрузки плейлиста, дополненная начальными нулями в соответствии с общей длиной плейлиста.
- `playlist_uploader` (строка): Полное имя загрузчика плейлиста
- `playlist_uploader_id` (строка): Псевдоним или идентификатор пользователя, загрузившего плейлист
- `playlist_channel` (строка): Отображаемое имя канала, загрузившего плейлист
- `playlist_channel_id` (строка): Идентификатор канала, загрузившего плейлист
- `webpage_url` (строка): URL-адрес веб-страницы с видео, который, если его указать в yt-dlp, должен снова выдать тот же результат.
- `webpage_url_basename` (строка): Базовое имя URL-адреса веб-страницы.
- `webpage_url_domain` (строка): Домен URL веб-страницы
- `original_url` (строка): URL-адрес, предоставленный пользователем (или такой же, как `webpage_url` для записей в плейлисте)
- `categories` (список): Список категорий, к которым относится видео.
- `tags` (список): Список тегов, присвоенных видео.
- `cast` (список): Список актеров

Все поля в форматах фильтрации также могут быть использованы.

Доступно для видео, которое относится к какой-либо логической главе или разделу:

- `chapter` (строка): Название или заголовок главы, к которой относится видео
- `chapter_number` (числовой): Номер главы, к которой относится видео.
- `chapter_id` (строка): Идентификатор главы, к которой относится видео

Доступно для видео, являющегося эпизодом какого-либо сериала или программы:

- `series` (строка): Название сериала или программы, к которой относится видеоэпизод
- `series_id` (строка): Идентификатор сериала или программы, к которой относится видеоэпизод
- `season` (строка): Название сезона, к которому относится видеоэпизод

- season_number (числовой): Номер сезона, к которому относится эпизод видео.
- season_id (строка): Идентификатор сезона, к которому относится эпизод видео
- episode (строка): Название видеоэпизода
- episode_number (числовой): Номер эпизода видео в сезоне.
- episode_id (строка): Идентификатор эпизода видео

Доступно для носителя, являющегося треком или частью музыкального альбома:

- track (строка): Название трека
- track_number (числовой): Номер трека в альбоме или на диске.
- track_id (строка): Идентификатор трека
- artists (список): Исполнитель(и) трека
- artist (строка): Исполнитель(и) трека; через запятую
- genres (список): Жанр(ы) трека
- genre (строка): Жанр(ы) трека; через запятую
- composers (список): Композитор(ы) произведения
- composer (строка): Композитор(ы) произведения; через запятую
- album (строка): Название альбома, к которому принадлежит трек
- album_type (строка): Тип альбома
- album_artists (список): Все артисты, принявшие участие в альбоме
- album_artist (строка): Все исполнители, появившиеся в альбоме; разделенные запятой
- disc_number (числовой): Номер диска или другого физического носителя, к которому принадлежит трек

Доступно только при использовании `--download-sections` и для `chapter:` префикса при использовании `--split-chapters` для видео с внутренними главами:

- section_title (строка): Название главы
- section_number (числовой): Номер главы в файле
- section_start (числовое): Время начала главы в секундах
- section_end (числовое): Время окончания главы в секундах.

Доступно только при использовании в `--print`:

- urls (строка): URL-адреса всех запрошенных форматов, по одному в каждой строке
- filename (строка): Имя видеофайла. Обратите внимание, что фактическое имя файла может отличаться
- formats_table (таблица): Таблица форматов видео, напечатанная `--list-formats`
- thumbnails_table (таблица): Таблица форматов миниатюр, напечатанная `--list-thumbnails`
- subtitles_table (таблица): Таблица форматов субтитров, напечатанная `--list-subs`
- automatic_captions_table (таблица): Таблица форматов автоматических субтитров, напечатанная `--list-subs`

Доступно только после загрузки видео (`post_process` / `after_move`):

- filepath: Фактический путь к загруженному видеофайлу

Доступно только в `--sponsorblock-chapter-title`:

- start_time (числовое): Время начала главы в секундах
- end_time (числовое): Время окончания главы в секундах.
- categories (список): категории SponsorBlock, к которым относится глава
- category (строка): Наименьшая категория SponsorBlock, к которой принадлежит глава
- category_names (список): Понятные названия категорий
- name (строка): Понятное название наименьшей категории
- type (строка): Тип действия SponsorBlock главы

Каждая вышеупомянутая последовательность при ссылке в выходном шаблоне будет заменена фактическим значением, соответствующим имени последовательности. Например, для `-o %(title)s-%(id)s.%(ext)s` и видео mp4 с заголовком `yt-dlp test video` и идентификатором `BaW_jenozKc` это приведет к `yt-dlp test video-BaW_jenozKc.mp4` созданию файла в текущем каталоге.

Примечание: некоторые последовательности не гарантированно присутствуют, поскольку они зависят от метаданных, полученных определенным экстрактором. Такие последовательности будут заменены значением-заполнителем, предоставленным с помощью `--output-na-placeholder` (`NA` по умолчанию).

Совет: просмотрите `-j` вывод, чтобы определить, какие поля доступны для конкретного URL-адреса.

Для числовых последовательностей можно использовать числовое форматирование ; например, `%(view_count)05d` результатом будет строка с количеством просмотров, дополненная нулями до 5 символов, как в `00042` .

Выходные шаблоны также могут содержать произвольный иерархический путь, например, `-o "%(playlist)s/%(playlist_index)s - %(title)s.%(ext)s"` который приведет к загрузке каждого видео в каталог, соответствующий этому шаблону пути. Любой отсутствующий каталог будет автоматически создан для вас.

Для использования процентных литералов в шаблоне вывода используйте `%%` . Для вывода в stdout используйте `-o -` .

Текущий шаблон по умолчанию — `%(title)s [%(id)s].%(ext)s` .

В некоторых случаях вам не нужны специальные символы, такие как 中, пробелы или &, например, при передаче загруженного имени файла в систему Windows или имени файла через 8-битный небезопасный канал. В этих случаях добавьте флаг, `--restrict-filenames` чтобы получить более короткое название.

Примеры выходных шаблонов

```
$ yt-dlp --print filename -o "test video.%(ext)s" BaW_jenozKc
test video.webm    # Literal name with correct extension

$ yt-dlp --print filename -o "%(title)s.%(ext)s" BaW_jenozKc
youtube-dl test video ''_ä↭𝕐.webm    # All kinds of weird characters

$ yt-dlp --print filename -o "%(title)s.%(ext)s" BaW_jenozKc --restrict-filenames
youtube-dl_test_video_.webm    # Restricted file name

# Download YouTube playlist videos in separate directory indexed by video order in a playlist
$ yt-dlp -o "%(playlist)s/%(playlist_index)s - %(title)s.%(ext)s" "https://www.youtube.com/playlist?list=PLwiyx1dc3P2JR9N8gQaQN_BCvlSl

# Download YouTube playlist videos in separate directories according to their uploaded year
$ yt-dlp -o "%(upload_date>%Y)s/%(title)s.%(ext)s" "https://www.youtube.com/playlist?list=PLwiyx1dc3P2JR9N8gQaQN_BCvlSlap7re"

# Prefix playlist index with " - " separator, but only if it is available
$ yt-dlp -o "%(playlist_index&{} - |)s%(title)s.%(ext)s" BaW_jenozKc "https://www.youtube.com/user/TheLinuxFoundation/playlists"

# Download all playlists of YouTube channel/user keeping each playlist in separate directory:
$ yt-dlp -o "%(uploader)s/%(playlist)s/%(playlist_index)s - %(title)s.%(ext)s" "https://www.youtube.com/user/TheLinuxFoundation/playli

# Download Udemy course keeping each chapter in separate directory under MyVideos directory in your home
$ yt-dlp -u user -p password -P "~/MyVideos" -o "%(playlist)s/%(chapter_number)s - %(chapter)s/%(title)s.%(ext)s" "https://www.udem

# Download entire series season keeping each series and each season in separate directory under C:/MyVideos
$ yt-dlp -P "C:/MyVideos" -o "%(series)s/%(season_number)s - %(season)s/%(episode_number)s - %(episode)s.%(ext)s" "https://videomore.r

# Download video as "C:\MyVideos\uploader\title.ext", subtitles as "C:\MyVideos\subs\uploader\title.ext"
# and put all temporary files in "C:\MyVideos\tmp"
$ yt-dlp -P "C:/MyVideos" -P "temp:tmp" -P "subtitle:subs" -o "%(uploader)s/%(title)s.%(ext)s" BaW_jenoz --write-subs

# Download video as "C:\MyVideos\uploader\title.ext" and subtitles as "C:\MyVideos\uploader\subs\title.ext"
$ yt-dlp -P "C:/MyVideos" -o "%(uploader)s/%(title)s.%(ext)s" -o "subtitle:%(uploader)s/subs/%(title)s.%(ext)s" BaW_jenozKc --write

# Stream the video being downloaded to stdout
$ yt-dlp -o - BaW_jenozKc
```

ВЫБОР ФОРМАТА

По умолчанию yt-dlp пытается загрузить наилучшее доступное качество, если вы не передадите никаких параметров. Это, как правило, эквивалентно использованию `-f bestvideo*+bestaudio/best` . Однако, если включено несколько аудиопотоков (`--audio-multistreams`), формат по умолчанию меняется на `-f bestvideo+bestaudio/best` . Аналогично, если ffmpeg недоступен или если вы используете yt-dlp для потоковой передачи `stdout` (`-o -`), форматом по умолчанию становится `-f best/bestvideo+bestaudio` .

Предупреждение об устаревании : последние версии yt-dlp могут одновременно передавать несколько форматов на stdout с помощью ffmpeg. Поэтому в будущих версиях значение по умолчанию для этого будет установлено `-f bv*+ba/b` аналогично обычным загрузкам. Если вы хотите сохранить настройку `-f b/bv+ba` , рекомендуется явно указать ее в параметрах конфигурации.

Общий синтаксис для выбора формата: `-f FORMAT` (или `--format FORMAT`), где `FORMAT` — *выражение селектора* , т. е. выражение, описывающее формат или форматы, которые вы хотите загрузить.

tl;dr: перенаправьте меня к примерам .

The simplest case is requesting a specific format; e.g. with `-f 22` you can download the format with format code equal to 22. You can get the list of available format codes for particular video using `--list-formats` or `-F`. Note that these format codes are extractor specific.

You can also use a file extension (currently `3gp`, `aac`, `flv`, `m4a`, `mp3`, `mp4`, `ogg`, `wav`, `webm` are supported) to download the best quality format of a particular file extension served as a single file, e.g. `-f webm` will download the best quality format with the `webm` extension served as a single file.

You can use `-f -` to interactively provide the format selector *for each video*

You can also use special names to select particular edge case formats:

- `all`: Select all formats separately
- `mergeall`: Select and merge all formats (Must be used with `--audio-multistreams`, `--video-multistreams` or both)
- `b*`, `best*`: Select the best quality format that contains either a video or an audio or both (i.e.; `vcodec!=none or acodec!=none`)
- `b`, `best`: Select the best quality format that contains both video and audio. Equivalent to `best*[vcodec!=none][acodec!=none]`
- `bv`, `bestvideo`: Select the best quality video-only format. Equivalent to `best*[acodec=none]`
- `bv*`, `bestvideo*`: Select the best quality format that contains video. It may also contain audio. Equivalent to `best*[vcodec!=none]`
- `ba`, `bestaudio`: Select the best quality audio-only format. Equivalent to `best*[vcodec=none]`
- `ba*`, `bestaudio*`: Select the best quality format that contains audio. It may also contain video. Equivalent to `best*[acodec!=none]` (Do not use!)
- `w*`, `worst*`: Select the worst quality format that contains either a video or an audio
- `w`, `worst`: Select the worst quality format that contains both video and audio. Equivalent to `worst*[vcodec!=none][acodec!=none]`
- `wv`, `worstvideo`: Select the worst quality video-only format. Equivalent to `worst*[acodec=none]`
- `wv*`, `worstvideo*`: Select the worst quality format that contains video. It may also contain audio. Equivalent to `worst*[vcodec!=none]`
- `wa`, `worstaudio`: Select the worst quality audio-only format. Equivalent to `worst*[vcodec=none]`
- `wa*`, `worstaudio*`: Select the worst quality format that contains audio. It may also contain video. Equivalent to `worst*[acodec!=none]`

For example, to download the worst quality video-only format you can use `-f worstvideo`. It is, however, recommended not to use `worst` and related options. When your format selector is `worst`, the format which is worst in all respects is selected. Most of the time, what you actually want is the video with the smallest filesize instead. So it is generally better to use `-S +size` or more rigorously, `-S +size,+br,+res,+fps` instead of `-f worst`. See Sorting Formats for more details.

You can select the n'th best format of a type by using `best<type>.<n>`. For example, `best.2` will select the 2nd best combined format. Similarly, `bv*.3` will select the 3rd best format that contains a video stream.

If you want to download multiple videos, and they don't have the same formats available, you can specify the order of preference using slashes. Note that formats on the left hand side are preferred; e.g. `-f 22/17/18` will download format 22 if it's available, otherwise it will download format 17 if it's available, otherwise it will download format 18 if it's available, otherwise it will complain that no suitable formats are available for download.

If you want to download several formats of the same video use a comma as a separator, e.g. `-f 22,17,18` will download all these three formats, of course if they are available. Or a more sophisticated example combined with the precedence feature: `-f 136/137/mp4/bestvideo,140/m4a/bestaudio`.

You can merge the video and audio of multiple formats into a single file using `-f <format1>+<format2>+...` (requires ffmpeg installed); e.g. `-f bestvideo+bestaudio` will download the best video-only format, the best audio-only format and mux them together with ffmpeg.

Deprecation warning: Since the *below* described behavior is complex and counter-intuitive, this will be removed and multistreams will be enabled by default in the future. A new operator will be instead added to limit formats to single audio/video

Unless `--video-multistreams` is used, all formats with a video stream except the first one are ignored. Similarly, unless `--audio-multistreams` is used, all formats with an audio stream except the first one are ignored. E.g. `-f bestvideo+best+bestaudio --video-multistreams --audio-multistreams` will download and merge all 3 given formats. The resulting file will have 2 video streams and 2 audio streams. But `-f bestvideo+best+bestaudio --no-video-multistreams` will download and merge only `bestvideo` and `bestaudio`. `best` is ignored since another format containing a video stream (`bestvideo`) has already been selected. The order of the formats is therefore important. `-f best+bestaudio --no-audio-multistreams` will download only `best` while `-f bestaudio+best --no-audio-multistreams` will ignore `best` and download only `bestaudio`.

Filtering Formats

You can also filter the video formats by putting a condition in brackets, as in `-f "best[height=720]"` (or `-f "[filesize>10M]"` since filters without a selector are interpreted as `best`).

The following numeric meta fields can be used with comparisons `<`, `<=`, `>`, `>=`, `=` (equals), `!=` (not equals):

- `filesize`: The number of bytes, if known in advance
- `filesize_approx`: An estimate for the number of bytes
- `width`: Width of the video, if known
- `height`: Height of the video, if known
- `aspect_ratio`: Aspect ratio of the video, if known
- `tbr`: Average bitrate of audio and video in kbps
- `abr`: Average audio bitrate in kbps
- `vbr`: Average video bitrate in kbps
- `asr`: Audio sampling rate in Hertz
- `fps`: Frame rate
- `audio_channels`: The number of audio channels
- `stretched_ratio`: `width:height` of the video's pixels, if not square

Also filtering work for comparisons `=` (equals), `^=` (starts with), `$=` (ends with), `*=` (contains), `~=` (matches regex) and following string meta fields:

- `url`: Video URL
- `ext`: File extension
- `acodec`: Name of the audio codec in use
- `vcodec`: Name of the video codec in use
- `container`: Name of the container format
- `protocol`: The protocol that will be used for the actual download, lower-case (`http`, `https`, `rtsp`, `rtmp`, `rtmpe`, `mms`, `f4m`, `ism`, `http_dash_segments`, `m3u8`, or `m3u8_native`)
- `language`: Language code
- `dynamic_range`: The dynamic range of the video
- `format_id`: A short description of the format
- `format`: A human-readable description of the format
- `format_note`: Additional info about the format
- `resolution`: Textual description of width and height

Any string comparison may be prefixed with negation `!` in order to produce an opposite comparison, e.g. `!*=` (does not contain). The comparand of a string comparison needs to be quoted with either double or single quotes if it contains spaces or special characters other than `._-`.

Note: None of the aforementioned meta fields are guaranteed to be present since this solely depends on the metadata obtained by the particular extractor, i.e. the metadata offered by the website. Any other field made available by the extractor can also be used for filtering.

Formats for which the value is not known are excluded unless you put a question mark (`?`) after the operator. You can combine format filters, so `-f "bv[height<=?720][tbr>500]"` selects up to 720p videos (or videos where the height is not known) with a bitrate of at least 500 kbps. You can also use the filters with `all` to download all formats that satisfy the filter, e.g. `-f "all[vcodec=none]"` selects all audio-only formats.

Format selectors can also be grouped using parentheses; e.g. `-f "(mp4,webm)[height<480]"` will download the best pre-merged mp4 and webm formats with a height lower than 480.

Sorting Formats

You can change the criteria for being considered the `best` by using `-S` (`--format-sort`). The general format for this is `--format-sort field1,field2...`.

The available fields are:

- `hasvid`: Gives priority to formats that have a video stream
- `hasaud`: Gives priority to formats that have an audio stream
- `ie_pref`: The format preference
- `lang`: The language preference
- `quality`: The quality of the format
- `source`: The preference of the source
- `proto`: Protocol used for download (`https`/`ftps` > `http`/`ftp` > `m3u8_native`/`m3u8` > `http_dash_segments` > `websocket_frag` > `mms`/`rtsp` > `f4f`/`f4m`)
- `vcodec`: Video Codec (`av01` > `vp9.2` > `vp9` > `h265` > `h264` > `vp8` > `h263` > `theora` > other)
- `acodec`: Audio Codec (`flac`/`alac` > `wav`/`aiff` > `opus` > `vorbis` > `aac` > `mp4a` > `mp3` > `ac4` > `eac3` > `ac3` > `dts` > other)
- `codec`: Equivalent to `vcodec,acodec`
- `vext`: Video Extension (`mp4` > `mov` > `webm` > `flv` > other). If `--prefer-free-formats` is used, `webm` is preferred.
- `aext`: Audio Extension (`m4a` > `aac` > `mp3` > `ogg` > `opus` > `webm` > other). If `--prefer-free-formats` is used, the order changes to `ogg` > `opus` > `webm` > `mp3` > `m4a` > `aac`
- `ext`: Equivalent to `vext,aext`
- `filesize`: Exact filesize, if known in advance
- `fs_approx`: Approximate filesize
- `size`: Exact filesize if available, otherwise approximate filesize
- `height`: Height of video
- `width`: Width of video
- `res`: Video resolution, calculated as the smallest dimension.
- `fps`: Framerate of video
- `hdr`: The dynamic range of the video (`DV` > `HDR12` > `HDR10+` > `HDR10` > `HLG` > `SDR`)
- `channels`: The number of audio channels
- `tbr`: Total average bitrate in kbps
- `vbr`: Average video bitrate in kbps
- `abr`: Average audio bitrate in kbps
- `br`: Average bitrate in kbps, `tbr`/`vbr`/`abr`
- `asr`: Audio sample rate in Hz

Deprecation warning: Many of these fields have (currently undocumented) aliases, that may be removed in a future version. It is recommended to use only the documented field names.

All fields, unless specified otherwise, are sorted in descending order. To reverse this, prefix the field with a `+`. E.g. `+res` prefers format with the smallest resolution. Additionally, you can suffix a preferred value for the fields, separated by a `:`. E.g. `res:720` prefers larger videos, but no larger than 720p and the smallest video if there are no videos less than 720p. For `codec` and `ext`, you can provide two preferred values, the first for video and the second for audio. E.g. `+codec:avc:m4a` (equivalent to `+vcodec:avc,+acodec:m4a`) sets the video codec preference to `h264` > `h265` > `vp9` > `vp9.2` > `av01` > `vp8` > `h263` > `theora` and audio codec preference to `mp4a` > `aac` > `vorbis` > `opus` > `mp3` > `ac3` > `dts`. You can also make the sorting prefer the nearest values to the provided by using `~` as the delimiter. E.g. `filesize~1G` prefers the format with filesize closest to 1 GiB.

The fields `hasvid` and `ie_pref` are always given highest priority in sorting, irrespective of the user-defined order. This behavior can be changed by using `--format-sort-force`. Apart from these, the default order used is: `lang,quality,res,fps,hdr:12,vcodec:vp9.2,channels,acodec,size,br,asr,proto,ext,hasaud,source,id`. The extractors may override this default order, but they cannot override the user-provided order.

Note that the default has `vcodec:vp9.2`; i.e. `av1` is not preferred. Similarly, the default for hdr is `hdr:12`; i.e. Dolby Vision is not preferred. These choices are made since DV and AV1 formats are not yet fully compatible with most devices. This may be changed in the future as more devices become capable of smoothly playing back these formats.

If your format selector is `worst`, the last item is selected after sorting. This means it will select the format that is worst in all respects. Most of the time, what you actually want is the video with the smallest filesize instead. So it is generally better to use `-f best -S +size,+br,+res,+fps`.

Tip: You can use the `-v -F` to see how the formats have been sorted (worst to best).

Format Selection examples

```
# Download and merge the best video-only format and the best audio-only format,
# or download the best combined format if video-only format is not available
$ yt-dlp -f "bv+ba/b"

# Download best format that contains video,
# and if it doesn't already have an audio stream, merge it with best audio-only format
$ yt-dlp -f "bv*+ba/b"

# Same as above
$ yt-dlp

# Download the best video-only format and the best audio-only format without merging them
# For this case, an output template should be used since
# by default, bestvideo and bestaudio will have the same file name.
$ yt-dlp -f "bv,ba" -o "%(title)s.f%(format_id)s.%(ext)s"

# Download and merge the best format that has a video stream,
# and all audio-only formats into one file
$ yt-dlp -f "bv*+mergeall[vcodec=none]" --audio-multistreams

# Download and merge the best format that has a video stream,
# and the best 2 audio-only formats into one file
$ yt-dlp -f "bv*+ba+ba.2" --audio-multistreams


# The following examples show the old method (without -S) of format selection
# and how to use -S to achieve a similar but (generally) better result

# Download the worst video available (old method)
$ yt-dlp -f "wv*+wa/w"

# Download the best video available but with the smallest resolution
$ yt-dlp -S "+res"

# Download the smallest video available
$ yt-dlp -S "+size,+br"



# Download the best mp4 video available, or the best video if no mp4 available
$ yt-dlp -f "bv*[ext=mp4]+ba[ext=m4a]/b[ext=mp4] / bv*+ba/b"

# Download the best video with the best extension
# (For video, mp4 > mov > webm > flv. For audio, m4a > aac > mp3 ...)
$ yt-dlp -S "ext"



# Download the best video available but no better than 480p,
# or the worst video if there is no video under 480p
$ yt-dlp -f "bv*[height<=480]+ba/b[height<=480] / wv*+ba/w"

# Download the best video available with the largest height but no better than 480p,
# or the best video with the smallest resolution if there is no video under 480p
$ yt-dlp -S "height:480"

# Download the best video available with the largest resolution but no better than 480p,
# or the best video with the smallest resolution if there is no video under 480p
# Resolution is determined by using the smallest dimension.
# So this works correctly for vertical videos as well
$ yt-dlp -S "res:480"



# Download the best video (that also has audio) but no bigger than 50 MB,
# or the worst video (that also has audio) if there is no video under 50 MB
$ yt-dlp -f "b[filesize<50M] / w"

# Download largest video (that also has audio) but no bigger than 50 MB,
# or the smallest video (that also has audio) if there is no video under 50 MB
$ yt-dlp -f "b" -S "filesize:50M"

# Download best video (that also has audio) that is closest in size to 50 MB
```

```
$ yt-dlp -f "b" -S "filesize~50M"



# Download best video available via direct link over HTTP/HTTPS protocol,
# or the best video available via any protocol if there is no such video
$ yt-dlp -f "(bv*+ba/b)[protocol^=http][protocol!*=dash] / (bv*+ba/b)"

# Download best video available via the best protocol
# (https/ftps > http/ftp > m3u8_native > m3u8 > http_dash_segments ...)
$ yt-dlp -S "proto"



# Download the best video with either h264 or h265 codec,
# or the best video if there is no such video
$ yt-dlp -f "(bv*[vcodec~='^((he|a)vc|h26[45])']+ba) / (bv*+ba/b)"

# Download the best video with best codec no better than h264,
# or the best video with worst codec if there is no such video
$ yt-dlp -S "codec:h264"

# Download the best video with worst codec no worse than h264,
# or the best video with best codec if there is no such video
$ yt-dlp -S "+codec:h264"



# More complex examples

# Download the best video no better than 720p preferring framerate greater than 30,
# or the worst video (still preferring framerate greater than 30) if there is no such video
$ yt-dlp -f "((bv*[fps>30]/bv*)[height<=720]/(wv*[fps>30]/wv*)) + ba / (b[fps>30]/b)[height<=720]/(w[fps>30]/w)"

# Download the video with the largest resolution no better than 720p,
# or the video with the smallest resolution available if there is no such video,
# preferring larger framerate for formats with the same resolution
$ yt-dlp -S "res:720,fps"



# Download the video with smallest resolution no worse than 480p,
# or the video with the largest resolution available if there is no such video,
# preferring better codec and then larger total bitrate for the same resolution
$ yt-dlp -S "+res:480,codec,br"
```

MODIFYING METADATA

The metadata obtained by the extractors can be modified by using `--parse-metadata` and `--replace-in-metadata`

`--replace-in-metadata FIELDS REGEX REPLACE` is used to replace text in any metadata field using Python regular expression. Backreferences can be used in the replace string for advanced use.

The general syntax of `--parse-metadata FROM:TO` is to give the name of a field or an output template to extract data from, and the format to interpret it as, separated by a colon `:`. Either a Python regular expression with named capture groups, a single field name, or a similar syntax to the output template (only `%(field)s` formatting is supported) can be used for `TO`. The option can be used multiple times to parse and modify various fields.

Note that these options preserve their relative order, allowing replacements to be made in parsed fields and viceversa. Also, any field thus created can be used in the output template and will also affect the media file's metadata added when using `--embed-metadata`.

This option also has a few special uses:

- You can download an additional URL based on the metadata of the currently downloaded video. To do this, set the field `additional_urls` to the URL that you want to download. E.g. `--parse-metadata "description:(?P<additional_urls>https?://www\.vimeo\.com/\d+)"` will download the first vimeo video found in the description
- You can use this to change the metadata that is embedded in the media file. To do this, set the value of the corresponding field with a `meta_` prefix. For example, any value you set to `meta_description` field will be added to the `description` field in the file - you can use this to set a different "description" and "synopsis". To modify the metadata of individual streams, use the `meta<n>_` prefix (e.g. `meta1_language`). Any value set to the `meta_` field will overwrite all default values.

Note: Metadata modification happens before format selection, post-extraction and other post-processing operations. Some fields may be added or changed during these steps, overriding your changes.

For reference, these are the fields yt-dlp adds by default to the file metadata:

Metadata fields	From
title	track or title
date	upload_date
description , synopsis	description
purl , comment	webpage_url
track	track_number
artist	artist , artists , creator , creators , uploader or uploader_id
composer	composer or composers
genre	genre or genres
album	album
album_artist	album_artist or album_artists
disc	disc_number
show	series
season_number	season_number
episode_id	episode or episode_id
episode_sort	episode_number
language of each stream	the format's language

Note: The file format may not support some of these fields

Modifying metadata examples

```
# Interpret the title as "Artist - Title"
$ yt-dlp --parse-metadata "title:%(artist)s - %(title)s"

# Regex example
$ yt-dlp --parse-metadata "description:Artist - (?P<artist>.+)"

# Set title as "Series name S01E05"
$ yt-dlp --parse-metadata "%(series)s S%(season_number)02dE%(episode_number)02d:%(title)s"

# Prioritize uploader as the "artist" field in video metadata
$ yt-dlp --parse-metadata "%(uploader|)s:%(meta_artist)s" --embed-metadata

# Set "comment" field in video metadata using description instead of webpage_url,
# handling multiple lines correctly
$ yt-dlp --parse-metadata "description:(?s)(?P<meta_comment>.+)" --embed-metadata

# Do not set any "synopsis" in the video metadata
$ yt-dlp --parse-metadata ":(?P<meta_synopsis>)"

# Remove "formats" field from the infojson by setting it to an empty string
$ yt-dlp --parse-metadata "video::(?P<formats>)" --write-info-json

# Replace all spaces and "_" in title and uploader with a `-`
$ yt-dlp --replace-in-metadata "title,uploader" "[ _]" "-"
```

EXTRACTOR ARGUMENTS

Some extractors accept additional arguments which can be passed using `--extractor-args KEY:ARGS` . `ARGS` is a `;` (semicolon) separated string of `ARG=VAL1,VAL2` . E.g. `--extractor-args "youtube:player-client=android_embedded,web;formats=incomplete" --extractor-args "funimation:version=uncut"`

Note: In CLI, `ARG` can use `-` instead of `_` ; e.g. `youtube:player-client"` becomes `youtube:player_client"`

The following extractors use this feature:

youtube

- `lang` : Prefer translated metadata (`title` , `description` etc) of this language code (case-sensitive). By default, the video primary language metadata is preferred, with a fallback to `en` translated. See youtube.py for list of supported content language codes
- `skip` : One or more of `hls` , `dash` or `translated_subs` to skip extraction of the m3u8 manifests, dash manifests and auto-translated subtitles respectively
- `player_client` : Clients to extract video data from. The main clients are `web` , `ios` and `android` , with variants `_music` , `_embedded` , `_embedscreen` , `_creator` (e.g. `web_embedded`); and `mediaconnect` , `mweb` , `mweb_embedscreen` and `tv_embedded` (agegate bypass) with no variants. By default, `ios,web` is used, but `tv_embedded` and `creator` variants are added as required for age-gated videos. Similarly, the music variants are added for `music.youtube.com` urls. The `android` clients will always be given lowest priority since their formats are broken. You can use `all` to use all the clients, and `default` for the default clients.
- `player_skip` : Skip some network requests that are generally needed for robust extraction. One or more of `configs` (skip client configs), `webpage` (skip initial webpage), `js` (skip js player). While these options can help reduce the number of requests needed or avoid some rate-limiting, they could cause some issues. See #860 for more details
- `player_params` : YouTube player parameters to use for player requests. Will overwrite any default ones set by yt-dlp.
- `comment_sort` : `top` or `new` (default) - choose comment sorting mode (on YouTube's side)
- `max_comments` : Limit the amount of comments to gather. Comma-separated list of integers representing `max-comments,max-parents,max-replies,max-replies-per-thread` . Default is `all,all,all,all`
  - E.g. `all,all,1000,10` will get a maximum of 1000 replies total, with up to 10 replies per thread. `1000,all,100` will get a maximum of 1000 comments, with a maximum of 100 replies total
- `formats` : Change the types of formats to return. `dashy` (convert HTTP to DASH), `duplicate` (identical content but different URLs or protocol; includes `dashy`), `incomplete` (cannot be downloaded completely - live dash and post-live m3u8)
- `innertube_host` : Innertube API host to use for all API requests; e.g. `studio.youtube.com` , `youtubei.googleapis.com` . Note that cookies exported from one subdomain will not work on others
- `innertube_key` : Innertube API key to use for all API requests
- `raise_incomplete_data` : `Incomplete Data Received` raises an error instead of reporting a warning

youtubetab (YouTube playlists, channels, feeds, etc.)

- `skip` : One or more of `webpage` (skip initial webpage download), `authcheck` (allow the download of playlists requiring authentication when no initial webpage is downloaded. This may cause unwanted behavior, see #1122 for more details)
- `approximate_date` : Extract approximate `upload_date` and `timestamp` in flat-playlist. This may cause date-based filters to be slightly off

generic

- `fragment_query` : Passthrough any query in mpd/m3u8 manifest URLs to their fragments if no value is provided, or else apply the query string given as `fragment_query=VALUE` . Note that if the stream has an HLS AES-128 key, then the query parameters will be passed to the key URI as well, unless the `key_query` extractor-arg is passed, or unless an external key URI is provided via the `hls_key` extractor-arg. Does not apply to ffmpeg
- `variant_query` : Passthrough the master m3u8 URL query to its variant playlist URLs if no value is provided, or else apply the query string given as `variant_query=VALUE`
- `key_query` : Passthrough the master m3u8 URL query to its HLS AES-128 decryption key URI if no value is provided, or else apply the query string given as `key_query=VALUE` . Note that this will have no effect if the key URI is provided via the `hls_key` extractor-arg. Does not apply to ffmpeg
- `hls_key` : An HLS AES-128 key URI *or* key (as hex), and optionally the IV (as hex), in the form of `(URI|KEY)[,IV]` ; e.g. `generic:hls_key=ABCDEF1234567980,0xFEDCBA0987654321` . Passing any of these values will force usage of the native HLS downloader and override the corresponding values found in the m3u8 playlist
- `is_live` : Bypass live HLS detection and manually set `live_status` - a value of `false` will set `not_live` , any other value (or no value) will set `is_live`

funimation

- `language` : Audio languages to extract, e.g. `funimation:language=english,japanese`
- `version` : The video version to extract - `uncut` or `simulcast`

crunchyrollbeta (Crunchyroll)

- `hardsub` : One or more hardsub versions to extract (in order of preference), or `all` (default: `None` = no hardsubs will be extracted), e.g. `crunchyrollbeta:hardsub=en-US,de-DE`

vikichannel

- video_types : Types of videos to download - one or more of episodes , movies , clips , trailers

niconico

- segment_duration : Segment duration in milliseconds for HLS-DMC formats. Use it at your own risk since this feature may result in your account termination.

youtubewebarchive

- check_all : Try to check more at the cost of more requests. One or more of thumbnails , captures

gamejolt

- comment_sort : hot (default), you (cookies needed), top , new - choose comment sorting mode (on GameJolt's side)

hotstar

- res : resolution to ignore - one or more of sd , hd , fhd
- vcodec : vcodec to ignore - one or more of h264 , h265 , dvh265
- dr : dynamic range to ignore - one or more of sdr , hdr10 , dv

niconicochannelplus

- max_comments : Maximum number of comments to extract - default is 120

tiktok

- api_hostname : Hostname to use for mobile API calls, e.g. api22-normal-c-alisg.tiktokv.com
- app_name : Default app name to use with mobile API calls, e.g. trill
- app_version : Default app version to use with mobile API calls - should be set along with manifest_app_version , e.g. 34.1.2
- manifest_app_version : Default numeric app version to use with mobile API calls, e.g. 2023401020
- aid : Default app ID to use with mobile API calls, e.g. 1180
- app_info : Enable mobile API extraction with one or more app info strings in the format of <iid>/[app_name]/[app_version]/[manifest_app_version]/[aid] , where iid is the unique app install ID. iid is the only required value; all other values and their / separators can be omitted, e.g. tiktok:app_info=1234567890123456789 or tiktok:app_info=123,456/trill///1180,789//34.0.1/340001
- device_id : Enable mobile API extraction with a genuine device ID to be used with mobile API calls. Default is a random 19-digit string

rokfinchannel

- tab : Which tab to download - one of new , top , videos , podcasts , streams , stacks

twitter

- api : Select one of graphql (default), legacy or syndication as the API for tweet extraction. Has no effect if logged in

stacommu, wrestleuniverse

- device_id : UUID value assigned by the website and used to enforce device limits for paid livestream content. Can be found in browser local storage

twitch

- client_id : Client ID value to be sent with GraphQL requests, e.g. twitch:client_id=kimne78kx3ncx6brgo4mv6wki5h1ko

nhkradirulive (NHK らじる★らじる LIVE)

- area : Which regional variation to extract. Valid areas are: sapporo , sendai , tokyo , nagoya , osaka , hiroshima , matsuyama , fukuoka . Defaults to tokyo

nflplusreplay

- type : Type(s) of game replays to extract. Valid types are: full_game , full_game_spanish , condensed_game and all_22 . You can use all to extract all available replay types, which is the default

jiocinema

- `refresh_token` : The `refreshToken` UUID from browser local storage can be passed to extend the life of your login session when logging in with `token` as username and the `accessToken` from browser local storage as password

jiosaavn

- `bitrate` : Audio bitrates to request. One or more of `16` , `32` , `64` , `128` , `320` . Default is `128,320`

afreecatvlive

- `cdn` : One or more CDN IDs to use with the API call for stream URLs, e.g. `gcp_cdn` , `gs_cdn_pc_app` , `gs_cdn_mobile_web` , `gs_cdn_pc_web`

soundcloud

- `formats` : Formats to request from the API. Requested values should be in the format of `{protocol}_{extension}` (omitting the bitrate), e.g. `hls_opus,http_aac` . The `*` character functions as a wildcard, e.g. `*_mp3` , and can be passed by itself to request all formats. Known protocols include `http` , `hls` and `hls-aes` ; known extensions include `aac` , `opus` and `mp3` . Original `download` formats are always extracted. Default is `http_aac,hls_aac,http_opus,hls_opus,http_mp3,hls_mp3`

orfon (orf:on)

- `prefer_segments_playlist` : Prefer a playlist of program segments instead of a single complete video when available. If individual segments are desired, use `--concat-playlist never --extractor-args "orfon:prefer_segments_playlist"`

bilibili

- `prefer_multi_flv` : Prefer extracting flv formats over mp4 for older videos that still provide legacy formats

digitalconcerthall

- `prefer_combined_hls` : Prefer extracting combined/pre-merged video and audio HLS formats. This will exclude 4K/HEVC video and lossless/FLAC audio formats, which are only available as split video/audio HLS formats

Note: These options may be changed/removed in the future without concern for backward compatibility

PLUGINS

Note that all plugins are imported even if not invoked, and that there are no checks performed on plugin code. Use plugins at your own risk and only if you trust the code!

Plugins can be of `<type>s` `extractor` or `postprocessor` .

- Extractor plugins do not need to be enabled from the CLI and are automatically invoked when the input URL is suitable for it.
- Extractor plugins take priority over built-in extractors.
- Postprocessor plugins can be invoked using `--use-postprocessor NAME` .

Plugins are loaded from the namespace packages `yt_dlp_plugins.extractor` and `yt_dlp_plugins.postprocessor` .

In other words, the file structure on the disk looks something like:

```
yt_dlp_plugins/
    extractor/
        myplugin.py
    postprocessor/
        myplugin.py
```

yt-dlp looks for these `yt_dlp_plugins` namespace folders in many locations (see below) and loads in plugins from all of them.

See the wiki for some known plugins

Installing Plugins

Plugins can be installed using various methods and locations.

1. Configuration directories: Plugin packages (containing a `yt_dlp_plugins` namespace folder) can be dropped into the following standard configuration locations:
    - User Plugins
        - `${XDG_CONFIG_HOME}/yt-dlp/plugins/<package name>/yt_dlp_plugins/` (recommended on Linux/macOS)
        - `${XDG_CONFIG_HOME}/yt-dlp-plugins/<package name>/yt_dlp_plugins/`
        - `${APPDATA}/yt-dlp/plugins/<package name>/yt_dlp_plugins/` (recommended on Windows)
        - `${APPDATA}/yt-dlp-plugins/<package name>/yt_dlp_plugins/`
        - `~/.yt-dlp/plugins/<package name>/yt_dlp_plugins/`
        - `~/yt-dlp-plugins/<package name>/yt_dlp_plugins/`
    - System Plugins
        - `/etc/yt-dlp/plugins/<package name>/yt_dlp_plugins/`
        - `/etc/yt-dlp-plugins/<package name>/yt_dlp_plugins/`
2. Executable location: Plugin packages can similarly be installed in a `yt-dlp-plugins` directory under the executable location (recommended for portable installations):
    - Binary: where `<root-dir>/yt-dlp.exe`, `<root-dir>/yt-dlp-plugins/<package name>/yt_dlp_plugins/`
    - Source: where `<root-dir>/yt_dlp/__main__.py`, `<root-dir>/yt-dlp-plugins/<package name>/yt_dlp_plugins/`
3. pip and other locations in `PYTHONPATH`
    - Plugin packages can be installed and managed using `pip`. See yt-dlp-sample-plugins for an example.
        - Note: plugin files between plugin packages installed with pip must have unique filenames.
    - Any path in `PYTHONPATH` is searched in for the `yt_dlp_plugins` namespace folder.
        - Note: This does not apply for Pyinstaller/py2exe builds.

`.zip`, `.egg` and `.whl` archives containing a `yt_dlp_plugins` namespace folder in their root are also supported as plugin packages.

- e.g. `${XDG_CONFIG_HOME}/yt-dlp/plugins/mypluginpkg.zip` where `mypluginpkg.zip` contains `yt_dlp_plugins/<type>/myplugin.py`

Run yt-dlp with `--verbose` to check if the plugin has been loaded.

Developing Plugins

See the yt-dlp-sample-plugins repo for a template plugin package and the Plugin Development section of the wiki for a plugin development guide.

All public classes with a name ending in `IE`/`PP` are imported from each file for extractors and postprocessors respectively. This respects underscore prefix (e.g. `_MyBasePluginIE` is private) and `__all__`. Modules can similarly be excluded by prefixing the module name with an underscore (e.g. `_myplugin.py`).

To replace an existing extractor with a subclass of one, set the `plugin_name` class keyword argument (e.g. `class MyPluginIE(ABuiltInIE, plugin_name='myplugin')` will replace `ABuiltInIE` with `MyPluginIE`). Since the extractor replaces the parent, you should exclude the subclass extractor from being imported separately by making it private using one of the methods described above.

If you are a plugin author, add yt-dlp-plugins as a topic to your repository for discoverability.

See the Developer Instructions on how to write and test an extractor.

EMBEDDING YT-DLP

yt-dlp makes the best effort to be a good command-line program, and thus should be callable from any programming language.

Your program should avoid parsing the normal stdout since they may change in future versions. Instead, they should use options such as `-J`, `--print`, `--progress-template`, `--exec` etc to create console output that you can reliably reproduce and parse.

From a Python program, you can embed yt-dlp in a more powerful fashion, like this:

```python
from yt_dlp import YoutubeDL

URLS = ['https://www.youtube.com/watch?v=BaW_jenozKc']
```

```
with YoutubeDL() as ydl:
    ydl.download(URLS)
```

Most likely, you'll want to use various options. For a list of options available, have a look at `yt_dlp/YoutubeDL.py` or `help(yt_dlp.YoutubeDL)` in a Python shell. If you are already familiar with the CLI, you can use `devscripts/cli_to_api.py` to translate any CLI switches to `YoutubeDL` params.

Tip: If you are porting your code from youtube-dl to yt-dlp, one important point to look out for is that we do not guarantee the return value of `YoutubeDL.extract_info` to be json serializable, or even be a dictionary. It will be dictionary-like, but if you want to ensure it is a serializable dictionary, pass it through `YoutubeDL.sanitize_info` as shown in the example below

Embedding examples

Extracting information

```
import json
import yt_dlp

URL = 'https://www.youtube.com/watch?v=BaW_jenozKc'

# ℹ See help(yt_dlp.YoutubeDL) for a list of available options and public functions
ydl_opts = {}
with yt_dlp.YoutubeDL(ydl_opts) as ydl:
    info = ydl.extract_info(URL, download=False)

    # ℹ ydl.sanitize_info makes the info json-serializable
    print(json.dumps(ydl.sanitize_info(info)))
```

Download using an info-json

```
import yt_dlp

INFO_FILE = 'path/to/video.info.json'

with yt_dlp.YoutubeDL() as ydl:
    error_code = ydl.download_with_info_file(INFO_FILE)

print('Some videos failed to download' if error_code
      else 'All videos successfully downloaded')
```

Extract audio

```
import yt_dlp

URLS = ['https://www.youtube.com/watch?v=BaW_jenozKc']

ydl_opts = {
    'format': 'm4a/bestaudio/best',
    # ℹ See help(yt_dlp.postprocessor) for a list of available Postprocessors and their arguments
    'postprocessors': [{  # Extract audio using ffmpeg
        'key': 'FFmpegExtractAudio',
        'preferredcodec': 'm4a',
    }]
}

with yt_dlp.YoutubeDL(ydl_opts) as ydl:
    error_code = ydl.download(URLS)
```

Filter videos

```
import yt_dlp

URLS = ['https://www.youtube.com/watch?v=BaW_jenozKc']

def longer_than_a_minute(info, *, incomplete):
    """Download only videos longer than a minute (or with unknown duration)"""
    duration = info.get('duration')
    if duration and duration < 60:
        return 'The video is too short'
```

```
ydl_opts = {
    'match_filter': longer_than_a_minute,
}

with yt_dlp.YoutubeDL(ydl_opts) as ydl:
    error_code = ydl.download(URLS)
```

Adding logger and progress hook

```
import yt_dlp

URLS = ['https://www.youtube.com/watch?v=BaW_jenozKc']

class MyLogger:
    def debug(self, msg):
        # For compatibility with youtube-dl, both debug and info are passed into debug
        # You can distinguish them by the prefix '[debug] '
        if msg.startswith('[debug] '):
            pass
        else:
            self.info(msg)

    def info(self, msg):
        pass

    def warning(self, msg):
        pass

    def error(self, msg):
        print(msg)


# i See "progress_hooks" in help(yt_dlp.YoutubeDL)
def my_hook(d):
    if d['status'] == 'finished':
        print('Done downloading, now post-processing ...')


ydl_opts = {
    'logger': MyLogger(),
    'progress_hooks': [my_hook],
}

with yt_dlp.YoutubeDL(ydl_opts) as ydl:
    ydl.download(URLS)
```

Add a custom PostProcessor

```
import yt_dlp

URLS = ['https://www.youtube.com/watch?v=BaW_jenozKc']

# i See help(yt_dlp.postprocessor.PostProcessor)
class MyCustomPP(yt_dlp.postprocessor.PostProcessor):
    def run(self, info):
        self.to_screen('Doing stuff')
        return [], info


with yt_dlp.YoutubeDL() as ydl:
    # i "when" can take any value in yt_dlp.utils.POSTPROCESS_WHEN
    ydl.add_post_processor(MyCustomPP(), when='pre_process')
    ydl.download(URLS)
```

Use a custom format selector

```
import yt_dlp

URLS = ['https://www.youtube.com/watch?v=BaW_jenozKc']

def format_selector(ctx):
    """ Select the best video and the best audio that won't result in an mkv.
    NOTE: This is just an example and does not handle all cases """
```

```
        # formats are already sorted worst to best
        formats = ctx.get('formats')[::-1]

        # acodec='none' means there is no audio
        best_video = next(f for f in formats
                          if f['vcodec'] != 'none' and f['acodec'] == 'none')

        # find compatible audio extension
        audio_ext = {'mp4': 'm4a', 'webm': 'webm'}[best_video['ext']]
        # vcodec='none' means there is no video
        best_audio = next(f for f in formats if (
            f['acodec'] != 'none' and f['vcodec'] == 'none' and f['ext'] == audio_ext))

        # These are the minimum required fields for a merged format
        yield {
            'format_id': f'{best_video["format_id"]}+{best_audio["format_id"]}',
            'ext': best_video['ext'],
            'requested_formats': [best_video, best_audio],
            # Must be + separated list of protocols
            'protocol': f'{best_video["protocol"]}+{best_audio["protocol"]}'
        }


ydl_opts = {
    'format': format_selector,
}

with yt_dlp.YoutubeDL(ydl_opts) as ydl:
    ydl.download(URLS)
```

CHANGES FROM YOUTUBE-DL

New features

- Forked from yt-dlc@f9401f2 and merged with youtube-dl@a08f2b7 (exceptions)
- SponsorBlock Integration: You can mark/remove sponsor sections in YouTube videos by utilizing the SponsorBlock API
- Format Sorting: The default format sorting options have been changed so that higher resolution and better codecs will be now preferred instead of simply using larger bitrate. Furthermore, you can now specify the sort order using `-S`. This allows for much easier format selection than what is possible by simply using `--format` (examples)
- Merged with animelover1984/youtube-dl: You get most of the features and improvements from animelover1984/youtube-dl including `--write-comments`, `BiliBiliSearch`, `BilibiliChannel`, Embedding thumbnail in mp4/ogg/opus, playlist infojson etc. Note that NicoNico livestreams are not available. See #31 for details.
- YouTube improvements:
  - Supports Clips, Stories (`ytstories:<channel UCID>`), Search (including filters)*, YouTube Music Search, Channel-specific search, Search prefixes (`ytsearch:`, `ytsearchdate:`)*, Mixes, and Feeds (`:ytfav`, `:ytwatchlater`, `:ytsubs`, `:ythistory`, `:ytrec`, `:ytnotif`)
  - Fix for n-sig based throttling *
  - Supports some (but not all) age-gated content without cookies
  - Download livestreams from the start using `--live-from-start` (*experimental*)
  - Channel URLs download all uploads of the channel, including shorts and live
- Cookies from browser: Cookies can be automatically extracted from all major web browsers using `--cookies-from-browser BROWSER[+KEYRING][:PROFILE][::CONTAINER]`
- Download time range: Videos can be downloaded partially based on either timestamps or chapters using `--download-sections`
- Split video by chapters: Videos can be split into multiple files based on chapters using `--split-chapters`
- Multi-threaded fragment downloads: Download multiple fragments of m3u8/mpd videos in parallel. Use `--concurrent-fragments` (`-N`) option to set the number of threads used
- Aria2c with HLS/DASH: You can use `aria2c` as the external downloader for DASH(mpd) and HLS(m3u8) formats
- New and fixed extractors: Many new extractors have been added and a lot of existing ones have been fixed. See the changelog or the list of supported sites
- New MSOs: Philo, Spectrum, SlingTV, Cablevision, RCN etc.

- Subtitle extraction from manifests: Subtitles can be extracted from streaming media manifests. See commit/be6202f for details
- Multiple paths and output templates: You can give different output templates and download paths for different types of files. You can also set a temporary path where intermediary files are downloaded to using `--paths (-P)`
- Portable Configuration: Configuration files are automatically loaded from the home and root directories. See CONFIGURATION for details
- Output template improvements: Output templates can now have date-time formatting, numeric offsets, object traversal etc. See output template for details. Even more advanced operations can also be done with the help of `--parse-metadata` and `--replace-in-metadata`
- Other new options: Many new options have been added such as `--alias`, `--print`, `--concat-playlist`, `--wait-for-video`, `--retry-sleep`, `--sleep-requests`, `--convert-thumbnails`, `--force-download-archive`, `--force-overwrites`, `--break-match-filter` etc
- Improvements: Regex and other operators in `--format`/`--match-filter`, multiple `--postprocessor-args` and `--downloader-args`, faster archive checking, more format selection options, merge multi-video/audio, multiple `--config-locations`, `--exec` at different stages, etc
- Plugins: Extractors and PostProcessors can be loaded from an external file. See plugins for details
- Self updater: The releases can be updated using `yt-dlp -U`, and downgraded using `--update-to` if required
- Automated builds: Nightly/master builds can be used with `--update-to nightly` and `--update-to master`

See changelog or commits for the full list of changes

Features marked with a * have been back-ported to youtube-dl

Differences in default behavior

Some of yt-dlp's default options are different from that of youtube-dl and youtube-dlc:

- yt-dlp supports only Python 3.8+, and *may* remove support for more versions as they become EOL; while youtube-dl still supports Python 2.6+ and 3.2+
- The options `--auto-number` (`-A`), `--title` (`-t`) and `--literal` (`-l`), no longer work. See removed options for details
- `avconv` is not supported as an alternative to `ffmpeg`
- yt-dlp stores config files in slightly different locations to youtube-dl. See CONFIGURATION for a list of correct locations
- The default output template is `%(title)s [%(id)s].%(ext)s`. There is no real reason for this change. This was changed before yt-dlp was ever made public and now there are no plans to change it back to `%(title)s-%(id)s.%(ext)s`. Instead, you may use `--compat-options filename`
- The default format sorting is different from youtube-dl and prefers higher resolution and better codecs rather than higher bitrates. You can use the `--format-sort` option to change this to any order you prefer, or use `--compat-options format-sort` to use youtube-dl's sorting order
- The default format selector is `bv*+ba/b`. This means that if a combined video + audio format that is better than the best video-only format is found, the former will be preferred. Use `-f bv+ba/b` or `--compat-options format-spec` to revert this
- Unlike youtube-dlc, yt-dlp does not allow merging multiple audio/video streams into one file by default (since this conflicts with the use of `-f bv*+ba`). If needed, this feature must be enabled using `--audio-multistreams` and `--video-multistreams`. You can also use `--compat-options multistreams` to enable both
- `--no-abort-on-error` is enabled by default. Use `--abort-on-error` or `--compat-options abort-on-error` to abort on errors instead
- When writing metadata files such as thumbnails, description or infojson, the same information (if available) is also written for playlists. Use `--no-write-playlist-metafiles` or `--compat-options no-playlist-metafiles` to not write these files
- `--add-metadata` attaches the `infojson` to `mkv` files in addition to writing the metadata when used with `--write-info-json`. Use `--no-embed-info-json` or `--compat-options no-attach-info-json` to revert this
- Some metadata are embedded into different fields when using `--add-metadata` as compared to youtube-dl. Most notably, `comment` field contains the `webpage_url` and `synopsis` contains the `description`. You can use `--parse-metadata` to modify this to your liking or use `--compat-options embed-metadata` to revert this
- `playlist_index` behaves differently when used with options like `--playlist-reverse` and `--playlist-items`. See #302 for details. You can use `--compat-options playlist-index` if you want to keep the earlier behavior
- The output of `-F` is listed in a new format. Use `--compat-options list-formats` to revert this

- Live chats (if available) are considered as subtitles. Use `--sub-langs all,-live_chat` to download all subtitles except live chat. You can also use `--compat-options no-live-chat` to prevent any live chat/danmaku from downloading
- YouTube channel URLs download all uploads of the channel. To download only the videos in a specific tab, pass the tab's URL. If the channel does not show the requested tab, an error will be raised. Also, `/live` URLs raise an error if there are no live videos instead of silently downloading the entire channel. You may use `--compat-options no-youtube-channel-redirect` to revert all these redirections
- Unavailable videos are also listed for YouTube playlists. Use `--compat-options no-youtube-unavailable-videos` to remove this
- The upload dates extracted from YouTube are in UTC when available. Use `--compat-options no-youtube-prefer-utc-upload-date` to prefer the non-UTC upload date.
- If `ffmpeg` is used as the downloader, the downloading and merging of formats happen in a single step when possible. Use `--compat-options no-direct-merge` to revert this
- Thumbnail embedding in `mp4` is done with mutagen if possible. Use `--compat-options embed-thumbnail-atomicparsley` to force the use of AtomicParsley instead
- Some internal metadata such as filenames are removed by default from the infojson. Use `--no-clean-infojson` or `--compat-options no-clean-infojson` to revert this
- When `--embed-subs` and `--write-subs` are used together, the subtitles are written to disk and also embedded in the media file. You can use just `--embed-subs` to embed the subs and automatically delete the separate file. See #630 (comment) for more info. `--compat-options no-keep-subs` can be used to revert this
- `certifi` will be used for SSL root certificates, if installed. If you want to use system certificates (e.g. self-signed), use `--compat-options no-certifi`
- yt-dlp's sanitization of invalid characters in filenames is different/smarter than in youtube-dl. You can use `--compat-options filename-sanitization` to revert to youtube-dl's behavior
- ~~yt-dlp tries to parse the external downloader outputs into the standard progress output if possible (Currently implemented: aria2c). You can use `--compat-options no-external-downloader-progress` to get the downloader output as-is~~
- yt-dlp versions between 2021.09.01 and 2023.01.02 applies `--match-filter` to nested playlists. This was an unintentional side-effect of 8f18ac and is fixed in d7b460. Use `--compat-options playlist-match-filter` to revert this
- yt-dlp versions between 2021.11.10 and 2023.06.21 estimated `filesize_approx` values for fragmented/manifest formats. This was added for convenience in f2fe69, but was reverted in 0dff8e due to the potentially extreme inaccuracy of the estimated values. Use `--compat-options manifest-filesize-approx` to keep extracting the estimated values
- yt-dlp uses modern http client backends such as `requests`. Use `--compat-options prefer-legacy-http-handler` to prefer the legacy http handler (`urllib`) to be used for standard http requests.
- The sub-modules `swfinterp`, `casefold` are removed.
- Passing `--simulate` (or calling `extract_info` with `download=False`) no longer alters the default format selection. See #9843 for details.

For ease of use, a few more compat options are available:

- `--compat-options all`: Use all compat options (Do NOT use this!)
- `--compat-options youtube-dl`: Same as `--compat-options all,-multistreams,-playlist-match-filter,-manifest-filesize-approx,-allow-unsafe-ext`
- `--compat-options youtube-dlc`: Same as `--compat-options all,-no-live-chat,-no-youtube-channel-redirect,-playlist-match-filter,-manifest-filesize-approx,-allow-unsafe-ext`
- `--compat-options 2021`: Same as `--compat-options 2022,no-certifi,filename-sanitization,no-youtube-prefer-utc-upload-date`
- `--compat-options 2022`: Same as `--compat-options 2023,playlist-match-filter,no-external-downloader-progress,prefer-legacy-http-handler,manifest-filesize-approx`
- `--compat-options 2023`: Currently does nothing. Use this to enable all future compat options

The following compat options restore vulnerable behavior from before security patches:

- `--compat-options allow-unsafe-ext`: Allow files with any extension (including unsafe ones) to be downloaded (GHSA-79w7-vh3h-8g4j)

  > ⚠️ Only use if a valid file download is rejected because its extension is detected as uncommon
  >
  > This option can enable remote code execution! Consider opening an issue instead!

Deprecated options

These are all the deprecated options and the current alternative to achieve the same effect

Almost redundant options

While these options are almost the same as their new counterparts, there are some differences that prevents them being redundant

```
  -j, --dump-json                 --print "%()j"
  -F, --list-formats              --print formats_table
  --list-thumbnails               --print thumbnails_table --print playlist:thumbnails_table
  --list-subs                     --print automatic_captions_table --print subtitles_table
```

Redundant options

While these options are redundant, they are still expected to be used due to their ease of use

```
  --get-description               --print description
  --get-duration                  --print duration_string
  --get-filename                  --print filename
  --get-format                    --print format
  --get-id                        --print id
  --get-thumbnail                 --print thumbnail
  -e, --get-title                 --print title
  -g, --get-url                   --print urls
  --match-title REGEX             --match-filter "title ~= (?i)REGEX"
  --reject-title REGEX            --match-filter "title !~= (?i)REGEX"
  --min-views COUNT               --match-filter "view_count >=? COUNT"
  --max-views COUNT               --match-filter "view_count <=? COUNT"
  --break-on-reject               Use --break-match-filter
  --user-agent UA                 --add-header "User-Agent:UA"
  --referer URL                   --add-header "Referer:URL"
  --playlist-start NUMBER         -I NUMBER:
  --playlist-end NUMBER           -I :NUMBER
  --playlist-reverse              -I ::-1
  --no-playlist-reverse           Default
  --no-colors                     --color no_color
```

Not recommended

While these options still work, their use is not recommended since there are other alternatives to achieve the same

```
  --force-generic-extractor       --ies generic,default
  --exec-before-download CMD      --exec "before_dl:CMD"
  --no-exec-before-download       --no-exec
  --all-formats                   -f all
  --all-subs                      --sub-langs all --write-subs
  --print-json                    -j --no-simulate
  --autonumber-size NUMBER        Use string formatting, e.g. %(autonumber)03d
  --autonumber-start NUMBER       Use internal field formatting like %(autonumber+NUMBER)s
  --id                            -o "%(id)s.%(ext)s"
  --metadata-from-title FORMAT    --parse-metadata "%(title)s:FORMAT"
  --hls-prefer-native             --downloader "m3u8:native"
  --hls-prefer-ffmpeg             --downloader "m3u8:ffmpeg"
  --list-formats-old              --compat-options list-formats (Alias: --no-list-formats-as-table)
  --list-formats-as-table         --compat-options -list-formats [Default] (Alias: --no-list-formats-old)
  --youtube-skip-dash-manifest    --extractor-args "youtube:skip=dash" (Alias: --no-youtube-include-dash-manifest)
  --youtube-skip-hls-manifest     --extractor-args "youtube:skip=hls" (Alias: --no-youtube-include-hls-manifest)
  --youtube-include-dash-manifest Default (Alias: --no-youtube-skip-dash-manifest)
  --youtube-include-hls-manifest  Default (Alias: --no-youtube-skip-hls-manifest)
  --geo-bypass                    --xff "default"
  --no-geo-bypass                 --xff "never"
  --geo-bypass-country CODE       --xff CODE
  --geo-bypass-ip-block IP_BLOCK  --xff IP_BLOCK
```

Developer options

These options are not intended to be used by the end-user

```
  --test                          Download only part of video for testing extractors
  --load-pages                    Load pages dumped by --write-pages
  --youtube-print-sig-code        For testing youtube signatures
  --allow-unplayable-formats      List unplayable formats also
  --no-allow-unplayable-formats   Default
```

Old aliases

These are aliases that are no longer documented for various reasons

```
  --avconv-location               --ffmpeg-location
  --clean-infojson                --clean-info-json
  --cn-verification-proxy URL     --geo-verification-proxy URL
  --dump-headers                  --print-traffic
  --dump-intermediate-pages       --dump-pages
  --force-write-download-archive  --force-write-archive
  --load-info                     --load-info-json
  --no-clean-infojson             --no-clean-info-json
  --no-split-tracks               --no-split-chapters
  --no-write-srt                  --no-write-subs
  --prefer-unsecure               --prefer-insecure
  --rate-limit RATE               --limit-rate RATE
  --split-tracks                  --split-chapters
  --srt-lang LANGS                --sub-langs LANGS
  --trim-file-names LENGTH        --trim-filenames LENGTH
  --write-srt                     --write-subs
  --yes-overwrites                --force-overwrites
```

Sponskrub Options

Support for SponSkrub has been deprecated in favor of the `--sponsorblock` options

```
  --sponskrub                     --sponsorblock-mark all
  --no-sponskrub                  --no-sponsorblock
  --sponskrub-cut                 --sponsorblock-remove all
  --no-sponskrub-cut              --sponsorblock-remove -all
  --sponskrub-force               Not applicable
  --no-sponskrub-force            Not applicable
  --sponskrub-location            Not applicable
  --sponskrub-args                Not applicable
```

No longer supported

These options may no longer work as intended

```
  --prefer-avconv                 avconv is not officially supported by yt-dlp (Alias: --no-prefer-ffmpeg)
  --prefer-ffmpeg                 Default (Alias: --no-prefer-avconv)
  -C, --call-home                 Not implemented
  --no-call-home                  Default
  --include-ads                   No longer supported
  --no-include-ads                Default
  --write-annotations             No supported site has annotations now
  --no-write-annotations          Default
  --compat-options seperate-video-versions  No longer needed
  --compat-options no-youtube-prefer-utc-upload-date  No longer supported
```

Removed

These options were deprecated since 2014 and have now been entirely removed

```
  -A, --auto-number               -o "%(autonumber)s-%(id)s.%(ext)s"
  -t, -l, --title, --literal      -o "%(title)s-%(id)s.%(ext)s"
```

CONTRIBUTING

See CONTRIBUTING.md for instructions on Opening an Issue and Contributing code to the project

WIKI

Более подробную информацию смотрите на Wiki.

Релизы 89

yt-dlp 2024.07.25 Последний
5 дней назад

+ 88 релизов

Спонсор этого проекта

https://github.com/yt-dlp/yt-dlp/blob/master/Collaborators.md#collaborators

Использован 55,1 тыс.


+ 55,064

Участники 1,329

+ 1315 участников

Языки

Питон 99,9% Другой 0,1%