

Bilibili_downloader

项目地址: https://github.com/laorange/bilibili_downloader/

[特点]

1. 使用了 pyside2 构建ui
2. 使用了 httpx 以协程方式下载
3. 借鉴了java接口式、面向对象开发 (java初学者的一次尝试😄)
4. (尽力了)准-全类型标注

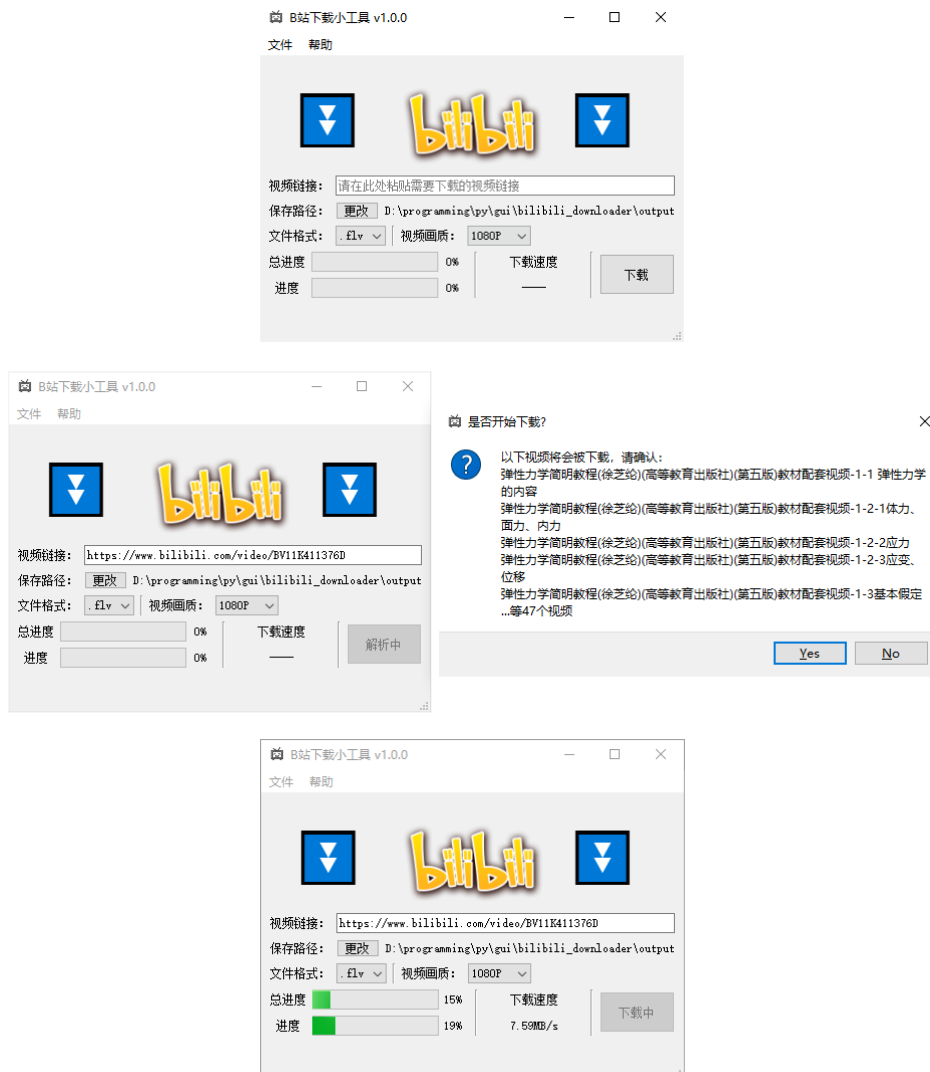
[功能]

- ☒ 能够下载bilibili单集的普通视频
- ☒ 能够下载bilibili多集的普通视频
- ☐ 能够下载bilibili番剧

[使用方法]

1. 在电脑浏览器上找到想要下载的b站视频, 复制网页链接
2. “视频链接”后的文本输入框内粘贴该链接
 - 下载**单集视频**或者**多集视频**的全部: 直接传入B站视频链接, 例: <https://www.bilibili.com/video/BV11K411376D>
 - 下载其中一集: 传入那一集的连接 (网址中会包含 ?p=), 例: <https://www.bilibili.com/video/BV11K411376D?p=8>
3. 点击“下载”按钮

[截图]



[二次开发-构思]

`VideoHandler` 在初始化时, 会调用 `get_proper_video_parser` 方法, 如下方代码所示

```
# video_handler.py
class VideoHandler:
    def __init__(self, url, quality: Union[str, int], video_format: str, save_path: Path):
        # ...
        self.video_parser: VideoParserInterface = self.get_proper_video_parser()

    def get_proper_video_parser(self) -> VideoParserInterface:
        if "bangumi" in self.url:
            return FanVideoParser(self.url, self.quality) # 为了下载番剧的解析器
        else:
            return NormalVideoParser(self.url, self.quality) # 默认的解析器
```

可以在此方法中自定义返回的视频分析器(继承自 `VideoParserInterface`)

```
# video_parsers.py
class VideoParserInterface(abc.ABC):
    def __init__(self, url: str, quality: Union[str, int]):
        # ...
        self.downloader_list: List[VideoDownloader] = self.get_downloader_list()

    @abc.abstractmethod
    def get_downloader_list(self) -> List[VideoDownloader]:
        pass
```

`downloader_list` 中的 `VideoDownloader`, 是下载器对象, 下载方法用的异步函数

```
# my_classes.py
class VideoDownloader:
    def __init__(self, title, page: PageInAPI):
        self.title = title
        self.page = page
        self.local_path = Path(__file__) # 这里是随便设个值, 反正后面要改

    async def download(self, save_path: Path, video_format: str = ".flv",
                      all_progress_value: Union[int, float] = 0, headers: dict = None):
        # ...
        with open(self.local_path / video_name, 'wb') as f:
            async with async_downloader.stream('GET', url) as response:
                async for chunk in response.aiter_bytes():
                    f.write(chunk)
```

`VideoDownloader.page` 的类型(`PageInAPI`)是用于记录某一集视频的详细信息(命名由来: b站的API对某一集的视频的名为 `Page`), 初始化参数 `info_dict` 是从api传来的json数据(转成字典后做实参传入)。特别说明, `VideoDownloader.title` 类似于电视剧剧名, 会作为文件夹的名字; 此处的 `PageInAPI.part` 类似于电视剧每一集的集名, 会作为视频的名称。

```
# my_classes.py
class PageInAPI:
    """用于记录api中的单个Page的信息"""
    def __init__(self, info_dict: Dict[str, Union[int, str]]):
        self.part = info_dict.get("part", '视频名')
        self.url: List[str] = []
        self.size: List[int] = []
        # ...其他属性非必需, 可以根据实际情况置空

    def set_url(self, url: str):
        self.url.append(url)

    def set_size(self, size: int = 1):
        self.size.append(size)
```

综上所述, 若需要扩展其他下载器, 或许可按以下步骤改写:

1. 自定义 视频分析器 (继承自 `VideoParserInterface`), 并重写 `get_downloader_list` 方法, 使其能返回 `List[VideoDownloader]`
2. 更改 `VideoHandler.get_proper_video_parser` 中的逻辑, 使其在指定条件下返回新的 视频分析器。

[近期修复的bug]

1.0.1

- ☒ 日志只在重启程序后才刷新的bug

1.0.0

- ☒ 包含 `?p=` 的链接解析错误
- ☒ 在下载期间退出窗口, 子线程仍在后台下载 或 主线程卡死无法退出
- ☒ 变更保存路径后, 需要重启才能恢复, 否则会报错

[参考]

- 下载视频的接口是参考了 [Henry](#) 的相关项目: [Henryhaohao/Bilibili_video_download](#)

[其他]

- 当前使用的视频解析器是不需要用户自行更新cookies的加密版本, **这是以牺牲下载速度换来的便利。**
- ui中的“主进度”其实是正在下载的文件序号占下载任务总数的百分比, 因此由于协程下载的特性, 会在各个文件反复横跳, 因此“主进度”仅用于提升观赏效果。
- 正因如此, 本程序下载多集视频的效率远高于下载单集视频。