

WhatsApp Scraper

Documentazione

Pollicelli Francesca, Palmisano Domenico

Github: <https://github.com/fpollicelli/whatsappScraper>

WhatsApp Scraper è un software realizzato in Python per lo scraping delle chat di WhatsApp tramite l’ausilio di WhatsApp Web.

Lo scraper è sviluppato come interfaccia grafica dalla quale è possibile gestire le impostazioni e ottenere un’anteprima dello scraping dei messaggi. Per ogni chat viene prodotto un file .csv contenente la data, l’ora, il mittente e il messaggio; le cartelle contenenti le chat e i media vengono zippate e ne viene generato un doppio hash tramite MD5 e SHA512.

Lo scraper non richiede installazione e può essere lanciato tramite l’eseguibile “WhatsAppScraper.exe”.

INDICE

Sommario

[**PREREQUISITI** 3](#_heading=h.gjdgxs)

[**Chromedriver – WebDriver per Chrome** 3](#_heading=h.30j0zll)

[**Chrome Browser** 3](#_heading=h.1fob9te)

[**LIBRERIE UTILIZZATE** 3](#_heading=h.3znysh7)

[**Selenium WebDriver** 3](#_heading=h.2et92p0)

[**TKinter** 3](#_heading=h.tyjcwt)

[**HashLib** 3](#_heading=h.3dy6vkm)

[**Threading** 3](#_heading=h.1t3h5sf)

[**Pyinstaller** 3](#_heading=h.4d34og8)

[**BACKEND** 4](#_heading=h.2s8eyo1)

[**FRONTEND** 4](#_heading=h.17dp8vu)

[**FUNZIONALITÀ** 5](#_heading=h.3rdcrjn)

[**Accesso tramite QR** 5](#_heading=h.26in1rg)

[**Scraping di ogni contatto** 5](#_heading=h.lnxbz9)

[**Scraping dei contatti archiviati** 5](#_heading=h.35nkun2)

[**Scraping da lista contatti** 5](#_heading=h.1ksv4uv)

[**Scraping media** 5](#_heading=h.44sinio)

[**Doppio hashing** 5](#_heading=h.2jxsxqh)

[**Scelta della cartella di destinazione** 6](#_heading=h.z337ya)

[**Generazione log** 6](#_heading=h.3j2qqm3)

[**Multilingua** 6](#_heading=h.1y810tw)

[**ISSUES** 7](#_heading=h.4i7ojhp)

[**Gestione dei caricamenti** 7](#_heading=h.2xcytpi)

[**SVILUPPI FUTURI** 8](#_heading=h.1ci93xb)

**PREREQUISITI**

## **Chromedriver – WebDriver per Chrome**

WebDriver è uno strumento open source per il test automatizzato di webapp su molti browser. Fornisce funzionalità per la navigazione nelle pagine Web, l'input dell'utente, l'esecuzione di JavaScript e altro ancora. ChromeDriver è un server autonomo che implementa lo standard W3C WebDriver.

Non è necessario scaricare Chromedriver in quanto è già compreso nella cartella contenente l’eseguibile.

## **Chrome Browser**

Attualmente, WhatsApp Scraper supporta unicamente il browser Chrome, aggiornato alla versione 88; è necessario installare Chrome per poter utilizzare lo scraper. <https://www.google.com/chrome/?brand=BNSD&gclsrc=ds&gclsrc=ds>

# **LIBRERIE UTILIZZATE**

## **Selenium WebDriver**

Selenium è una suite open source per la gestione automatizzata dei browser, utilizzata come framework di testing.

Selenium WebDriver è lo strumento che simula il comportamento di un utente reale all'interno di un browser: viene utilizzato per eseguire localmente o su macchine remote i test all'interno dei browser supportati.

## **TKinter**

Tkinter è la libreria che permette di creare interfacce grafiche nella programmazione con Python. Deriva dalle librerie Tcl/Tk ed è la libreria standard di questo linguaggio. WhatsApp Scraper utilizza TKinter per la creazione dell’interfaccia grafica e di tutte le sue componenti.

## **HashLib**

HashLib implementa algoritmi di hash e digest. Sono inclusi gli algoritmi di hash protetto SHA256, SHA384 e SHA512, nonché l'algoritmo MD5.

WhatsApp Scraper utilizza SHA512 e MD5 per generare un doppio hash della cartella contenente le chat e quella dei media scaricati.

## **Threading**

Il modulo Threading costruisce interfacce di thread di livello superiore sul modulo \_thread di livello inferiore. Viene utilizzato per la gestione contemporanea dell’interfaccia grafica e del browser Chrome.

## **Pyinstaller**

PyInstaller raggruppa un'applicazione Python e tutte le sue dipendenze in un unico pacchetto. L'utente può eseguire l'app nel pacchetto senza dover installare un interprete Python o alcun modulo, facilitando la distribuzione dello scraper.

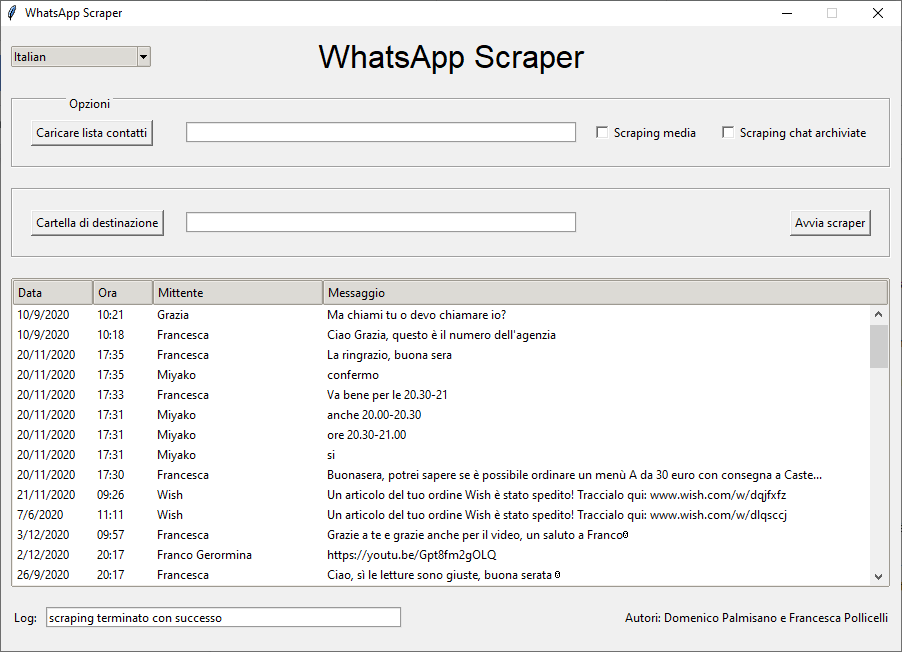
# **BACKEND**

Il backend di WhatsApp Scraper si fonda sull’utilizzo di Selenium WebDriver e del linguaggio XPath, il quale permette di individuare i nodi all'interno di un documento XML.

Per ogni funzionalità è stata programmata una funzione che automatizza le azioni che l’utente dovrebbe compiere per realizzarla: prendendo come esempio lo scraping di tutti i messaggi scambiati con un contatto specifico, l’utente dovrebbe aprire WhatsApp Web, cercare il contatto nella lista delle chat, cliccare sul contatto, scorrere tutta la lista dei messaggi fino a raggiungere il primo e salvare i messaggi in ordine cronologico. Questa operazione viene automatizzata in ogni sua fase: chromedriver aprirà WhatsApp Web in modo autonomo, il contatto verrà selezionato tramite XPath e, sempre tramite XPath, ogni messaggio verrà letto e salvato in un file .csv in ordine cronologico.

# **FRONTEND**

Il frontend è un’interfaccia grafica realizzata tramite libreria TKinter. L’obiettivo dell’interfaccia è quello di fornire un’anteprima del risultato di scraping e restituire dei feedback all’utente tramite log. Dall'interfaccia è possibile configurare le opzioni dello scraper, ovvero scegliere se salvare anche i media e dove salvarli, se effettuare lo scraping solo di una lista ristretta di contatti o se eseguire lo scraping di tutte le chat, eventualmente anche quelle archiviate. Vi è inoltre la possibilità di settare la lingua dell’interfaccia stessa.



# **FUNZIONALITÀ**

## **Accesso tramite QR**

L’accesso avviene tramite scannerizzazione di un codice QR, basandosi su quello della piattaforma Whatsapp Web. Questo protegge l’utente da dover utilizzare un id e una password e dal rischio che qualcuno usi e visualizzi le sue conversazioni in quanto è necessaria la presenza fisica dell’utente ma soltanto per la fase di accesso.

## **Scraping di ogni contatto**

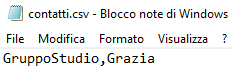
Rappresenta il core dell’applicazione, effettua lo scraping di tutti i messaggi di ogni chat principale, creando per ogni chat un file .csv con data di invio del messaggio, orario, mittente e contenuto del messaggio, comprensivo di emoji. Questa funzione è in grado di gestisce i contatti singoli, i gruppi e anche le liste broadcast.

## **Scraping dei contatti archiviati**

Spuntando l’apposito checkbox nell’interfaccia grafica, l’applicazione sposterà automaticamente tutte le chat archiviate nella sezione principale, al fine di richiamare la funzione di scraping precedente. Una volta terminato lo scraping, le chat de-archiviate verranno spostate nuovamente in archivio.

## **Scraping da lista contatti**

È possibile scegliere una lista di contatti tramite file .csv contenente i nomi delle chat da processare.  
Lo scraping da lista contatti gestisce anche le chat archiviate: se non sono presenti fra le chat principali, vengono ricercate in archivio e gestite come nella funzione di scraping di contatti archiviati.  
Non è necessario inserire emoji nel file, le chat con questo tipo di caratteri speciali verranno selezionate correttamente anche omettendoli.



## **Scraping media**

Spuntando l’apposito checkbox nell’interfaccia grafica, l’applicazione salverà le immagini, i video, gli audio, i vocali, le gif e i documenti scambiati nelle chat.

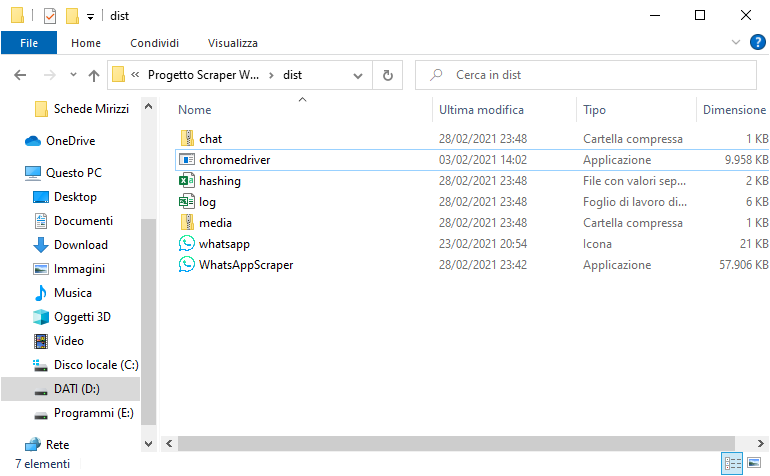
Nota: se il media non è più disponibile (poichè cancellato dal telefono), non potrà essere scaricato.

## **Doppio hashing**

Per la cartella contenente le chat e quella contenente i media viene generato un doppio hash tramite algoritmi MD5 e SHA512. Gli hash vengono salvati in due file (log.xls e hashing.csv, il primo utilizzato per una miglior visualizzazione dei dati, il secondo per essere utilizzato in applicazioni che accettano file di tipo csv) contenenti il percorso del file, il timestamp comprensivo di data, ora e fuso orario, il valore MD5 e quello SHA512.

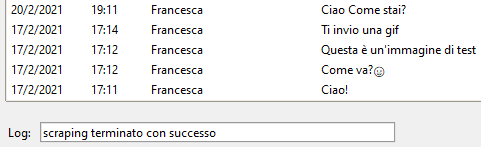
## **Scelta della cartella di destinazione** Tramite l’interfaccia è possibile scegliere una destinazione in cui salvare le chat e i media scaricati. Se la destinazione non viene selezionata, i file verranno salvati di default all’interno della cartella contenente l’eseguibile.

Terminato lo scraping, la cartella di progetto (o quella selezionata dall’utente) verrà aperta automaticamente.

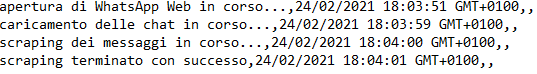


## **Generazione log**

Durante lo scraping è possibile consultare il log dall’interfaccia grafica. Il log viene mostrato per segnalare eventuali errori nello scraping e per notificare l’utente i progressi e le fasi del processo.



A scraping concluso, i messaggi di log vengono salvati nei file log.xsl e hashing.csv, comprensivi di timestamp con data, orario e fuso orario.



## **Multilingua**

WhatsApp Scraper effettua lo scraping dei messaggi tenendo conto della lingua impostata su WhatsApp Web (la quale coincide con la lingua del telefono dell’utilizzatore).



Questa gestione si rende necessaria non solo per la corretta selezione degli XPath, ma anche nello scraping delle date di invio dei messaggi, poiché vengono fornite in due formati diversi a seconda che la lingua sia l’italiano o l’inglese.

Italiano   Inglese

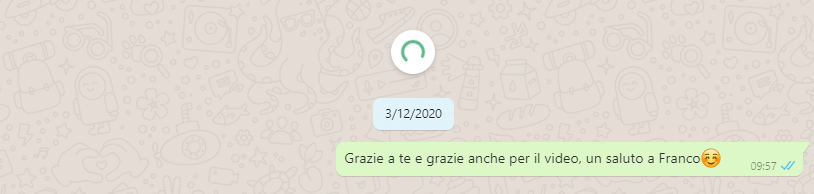
Indipendentemente dalla lingua del telefono, è possibile modificare quella dell’interfaccia grafica e dei messaggi di log, scegliendo fra italiano o inglese.

# **ISSUES**

## **Gestione dei caricamenti**

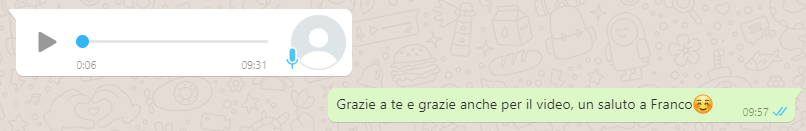
Poiché il codice si fonda sugli XPath degli elementi presenti in WhatsApp Web, la corretta gestione dei caricamenti è di fondamentale importanza.

Prendendo come esempio lo scraping dei messaggi di un contatto, dovremo attendere che tutti i messaggi siano stati caricati per poter procedere con lo scraping, semplicemente perchè, altrimenti, l’XPath del prossimo messaggio non sarebbe disponibile.



Per risolvere questo problema, viene utilizzata la classe WebDriverWait fornita da Selenium. Per ogni XPath, si attenderà finché l’elemento con tale XPath non verrà localizzato. Superato un certo limite di tempo, verrà sollevata un’eccezione.





Nonostante questa gestione conveniente dei caricamenti, alcune connessioni potrebbero richiedere tempi d’attesa più lunghi e generare errori anche se l’elemento sarà presente a breve.

# **SVILUPPI FUTURI**

Le lingue attualmente supportate sono l’italiano e l’inglese, ma è possibile aggiungerne agilmente di nuove modificando gli XPath. Un ulteriore miglioramento può essere applicato al browser, in quanto il software è attualmente utilizzabile unicamente con Google Chrome, ma potrebbe essere reso compatibile con altri browser tra i più utilizzati in commercio tramite Selenium WebDriver.

Un miglioramento riguardante l’implementazione del codice sorgente invece, potrebbe essere lo scraping dal basso delle chat in quanto renderebbe l’output visivo più rapido per l’utente che osserva l’esecuzione del software, un’anteprima più comprensibile e un’esecuzione più rapida non dovendo attendere il raggiungimento del primo messaggio della conversazione prima di iniziare la fase di scraping che potrebbe essere portata avanti man mano che si carica la chat dal basso.