# ParlaConIO

## Requisiti di contesto

Onboarding su app IO.

Pubblicazione di un servizio IO (backoffice IO).

Autorizzazione dell'IP da cui si chiamano i web service di IO (backoffice IO).

APIKEY del servizio IO.

## Requisiti di sistema

Python installato.

Modulo Python requests (py -m pip install requests).

I file del repository parlaConIO tutti in una cartella.

Compilare il file serviziIO.py con: un codice identificativo del servizio, le due APIKEY, il nome esteso del servizio, indicazioni circa la possibilità di inviare avvisi di pagamento con quel servizio (vincolo da backoffice IO) e di inviare messaggi con testo fisso (preferenza organizzativa).

Compilare il file serviziDiIncasso.py con i servizi di incasso attivi nella propria implementazione di PagoPA (se non si usa PagoPA o non si intendono veicolare avvisi di pagamento inserire almeno un servizio fittizio).

Consigliato: un editor di testi con evidenziazione della sintassi per formati JSON e Python (es.: Notepad++ con plugin JSON in ambienti WIndows); un software di foglio elettronico per lavorare su file CSV esternamente a parlaConIO (es.: LibreOffice).

Esempio: va benissimo un PC con Windows, installare Python con il suo installer, aggiungere requests e copiare i file del repository GitHub in una cartella (tipo C:\parlaConIO\).

## Consigli generali

In generale i dati per interagire con i web service di IO si passano tramite dei file CSV. Nel repository ci sono esempi minimali.

Conviene sempre fare delle prove con il codice fiscale di prova AAAAAA00A00A000A e magari con qualcuno decisamente inesistente (es.: AAZAAZ00Z00A000A) e "vedere l'effetto che fa" nella cosiddetta cartella di lotto.

Infatti, a ogni utilizzo, parlaConIO crea una "cartella di lotto" con nome composto da TIMESTAMP + funzione usata + servizio IO. I file di output e di log sono memorizzati in quella cartella.

Gli script del tipo inviaLotto*, dove possibile, guidano nell'associare le etichette delle colonne del CSV alle variabili presenti nel testo del messaggio. Alcuni script riconoscno i CSV conformi ai CSV presenti come esempio nel repository e, in tal caso, consentono di confermare l'associazione o definirne una nuova.

## Invio di lotto di messaggio

Selezionare lo script di tipo inviaLotto* adatto e seguire le istruzioni a video.

Verificare poi quanto prodotto nella cartella di lotto. In particolare il programma crea due file CSV derivati dal file dato in input che raccolgono:

- righe del CSV formalmente non corrette;

- righe del CSV non elaborate a causa del sovraccarico dei server IO (ipotesi teorica, mai verificatasi fin qui).

È possibile quindi correggere il primo CSV, unirlo al secondo (senza ripetere la prima riga) e lanciare l'invio di un nuovo lotto per completare l'invio precedente.

## Per fare verifiche su iscrizione di codici fiscali

Due opzioni:

1. solo codice fiscale: fornire un CSV in cui una colonna contiene il codice fiscale (lo script chiede quale colonna usare)

2. codice fiscale e nucleo familiare: fornire un CSV in cui una colonna contiene il codice fiscale e un'altra l'identificativo del nucleo familiare (lo script chiede di specificare entrambe le colonne).

I file CSV con i codici fiscali (e gli id dei nuclei familiari) conviene estrali dall'anagrafe comunale (per diffusione su popolazione – magari escludere dall'estrazione i minorenni) o da applicativi (se si vuole controllare la diffusione fra gli interessati a un dato servizio).

Rivoli, 3 ottobre 2021