Manuale d’installazione e uso

del sistema Butler

# Sommario

[1 Sommario 1](#_Toc72929534)

[2 Informazioni generali 1](#_Toc72929535)

[3 File forniti 1](#_Toc72929536)

[4 Librerie e requisiti comuni 2](#_Toc72929537)

[4.1 Ulteriori dipendenze in linux 2](#_Toc72929538)

[4.2 Requisiti per il centro di controllo 2](#_Toc72929539)

[4.3 Sicurezza 2](#_Toc72929540)

[5 Installazione e avvio automatizzato 2](#_Toc72929541)

[5.1 Windows 2](#_Toc72929542)

[5.2 Linux 2](#_Toc72929543)

[6 Database del server 3](#_Toc72929544)

[7 Riassunto delle informazioni 3](#_Toc72929545)

[8 File di configurazione 4](#_Toc72929546)

[9 Avvio manuale 6](#_Toc72929547)

[10 Uso del programma client 6](#_Toc72929548)

[11 Uso del programma server 6](#_Toc72929549)

[11.1 Nota 6](#_Toc72929550)

# Informazioni generali

Le seguenti istruzioni spiegano l’installazione, la configurazione e l’uso generale dei due programmi del sistema Butler, sia in Windows che in Linux. Si tratta quindi di una guida comprensiva per entrambi i software che, focalizzandosi sull’installazione, è pensata per gli amministratori e non va quindi consegnata agli utenti. La parte client e quella server funzionano in modo simile e sono forniti in un’unica cartella, ma hanno alcune dipendenze differenti e funzionano in modo opposto.

# File forniti

Per entrambi i programmi è fornito il codice sorgente, non compilato. Sarà compito di un amministratore occuparsi dell’installazione che consiste nel creare una cartella per il programma, creare un collegamento per l’avvio e posizionare il file di avvio automatico in una cartella apposita, se desiderato.

I file del progetto hanno la seguente struttura (sono mostrati solo i più importanti):

butler/

|---src/

|---client/

|----config.json

|----butler.dekstop

|---server/

|----config.json

|----butler.dekstop

|---common/

|---<altri files>

Le cartelle principali sono **client/**, **server/** (corrispondenti ai due programmi) e **common/** (file necessari ad entrambi). Questa andrà leggermente modificata a seconda del programma installato, come descritto nei prossimi capitoli.

# Librerie e requisiti comuni

I programmi necessitano **python 3.7.x** oppure **3.8.x**. Non sono testati con versioni precedenti e non funzionano con la versione 3.9.x.

I moduli aggiuntivi usati da entrambi i programmi possono essere scaricati con i comandi pip seguenti:

pip install PySimpleGUI PySimpleGUIWeb plyer pillow flask requests authlib pycryptodome

Per il programma **server** sarà necessario aggiungere anche:

pip install pymongo

Mentre per il **client**:

pip install PySimpleGUIQt psutil

Le differenze sono minime: il client usa **PySimpleGUIQt** per l’icona sulla taskbar (non disponibile completamente da PySimpleGUI standard basato su tkinter) e **psutil** per ricavare le informazioni, mentre il server necessiterà di **pymongo** per comunicare con il database.

## Ulteriori dipendenze in linux

A dipendenza della versione di python installata, potrebbe essere necessario aggiungere a parte anche **python3-tk** usando il gestore di pacchetti di sistema. Ad esempio, nelle distribuzioni basate di Ubuntu bisognerebbe eseguire:

sudo apt install python3-tk

## Requisiti per il centro di controllo

Il funzionamento dell’interfaccia grafica è assicurato su Microsoft Edge (versione chromium) e Firefox. Non sono assicurate tutte le caratteristiche su altri browser, nonostante non ci dovrebbero essere problemi con nessuna offerta moderna.

## Sicurezza

A dipendenza del firewall usato e delle configurazioni scelte, è necessario permettere la comunicazione tra server e client su una specifica porta (di default la 20210 in entrata sul server e la 20219 verso il client, dato che i due canali sono separati).

# Installazione e avvio automatizzato

## Windows

Sia per il programma client che per quello server è necessario creare una cartella nel percorso C:/Program Files. Il nome non è importante, ma deve contenere **src/** (quella fornita con i file del programma) dal quale si avrà semplicemente **rimosso il percorso superfluo**: se si sta installando l’agent si toglierà **server/**, mentre per installare il server si cancellerà **client/**.

Per l’installazione come servizio (avvio automatico al login), sia src/client che src/server contengono un file **app.pyw**. Di questo va incollato un collegamento (non il vero file) nella cartella C:\ProgramData\Microsoft\Windows\Start Menu\Programs\StartUp. È molto comodo per automatizzare l’avvio della parte client, ma potrebbe essere usato anche per il server.

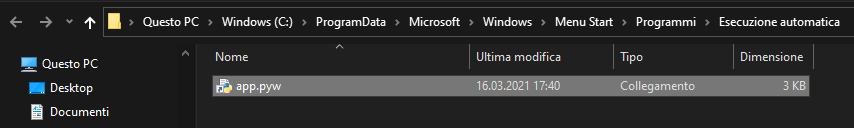


Figura 1 - La configurazione del file di avvio in Windows

## Linux

La cartella di installazione consigliata è /opt (che dovrebbe contenere i programmi esterni al sistema operativo). La struttura che deve contenere è uguale a quella in Windows: una cartella preferibilmente chiamata butler con all’interno **src/** senza la cartella dell’altro software.

Per avviare uno dei programmi automaticamente bisogna mettere una copia, questa volta reale, del file butler.desktop (presente per entrambi i software) nella cartella /etc/xdg/autostart e assicurarsi che i permessi sull’intera struttura a partire da /opt/butler permetta all’entità **other** di eseguire i programmi (in caso python non sia installato a livello di root). Se non si usa la cartella di installazione /opt/butler, va modificata la voce Exec del file butler.desktop).

I permessi minimi necessari per eseguire il file butler.dektop sono 005 (**read** ed **execute** a **other**), mentre per permettere poi di avviare il programma, **butler.py/manager.py** devono essere impostati su 007 (dato che bisogna sia leggere il file, sia scrivere nei log, sia eseguire il programma).

# Database del server

Un’importante componente per il server (assente per il client) è il database. È infatti necessaria un’istanza di MongoDB (il programma è testato sulla 4.4.3), nel quale va creato il database **butler** con tre collections: **buffer**, **notification** e **computer**. Queste ultime sono create automaticamente dal programma se non presenti.

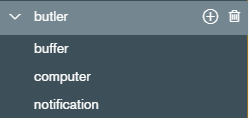


Figura 2 - Il database MongoDB e le tre collections

Un altro requisito importante, prima di poter usare il programma server, è l’aggiunta manuale della notifica “” a notification. Il file allegato “default\_notif.json” contiene i parametri standard da usare, ma si possono modificare se necessario. Si tratta del “template” dal quale vengono presi i valori (di base perlopiù vuoti) che verranno usati come **notifica vuota** nell’interfaccia grafica: i suoi attributi permettono di resettare i campi di input allo sstato originale. Non può essere sovrascritta dal centro di controllo. Non deve trattarsi di un template da inviare, ma di valori possibilmente vuoti dai quali si potranno costruire tutte le altre notifiche.

# Riassunto delle informazioni

Di seguito un riassunto delle informazioni più importanti a colpo d’occhio:

|  |  |  |
| --- | --- | --- |
|  | Windows | Linux |
| Percorso d’installazione consigliato | C:/Program Files/butler | /opt/butler |
| File necessari per il client | Tutti i file in src/ tranne src/server | |
| File necessari per il server | Tutti i file in src/ tranne src/client | |
| Avvio manuale del client | client/butler.py | |
| Avvio manuale del server | server/manager.py | |
| Nome del file di avvio | app.pyw | butler.desktop |
| Posizione del file di avvio | C:\ProgramData\Microsoft\Windows\Start Menu\Programs\StartUp | /etc/xdg/autostart/ |

# File di configurazione

I file **config.json** presenti sia in client/ che in server/ hanno una sintassi comune e vari parametri uguali. Nonostante siano teoricamente intercambiabili e le voci non necessarie siano semplicemente ignorate, è fondamentale che appaiano tutte quelle descritte di seguito (in base alla presenza di una X sulla colonna corrispondente al programma). Questi parametri sono ora elencati insieme ad una descrizione del loro scopo, i valori accettati e i valori impostati di default:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Configurazione | Spiegazione | Valore possibile | Default (se rilevante) | Client | Server |
| expireTime | il tempo di durata dei token, in minuti | Numero intero positivo | 10 | X | X |
| protocol | il protocollo (http o https) da usare per tutte le comunicazioni | “http” oppure “https” | "https" | X | X |
| ssl 🡪 certPath | Il percorso e il nome del certificato SSL | Percorso di sistema | "../butler.cert" | X | X |
| ssl 🡪 keyPath | Il percorso e il nome della chiave SSL | Percorso di sistema | "../butler.key" | X | X |
| server 🡪 ip | L’indirizzo sul quale contattare il server | IPv4 | - | X | X |
| server 🡪 port | La porta sulla quale contattare il server | Numero intero positivo | 20210 | X | X |
| server 🡪 id | Un identificativo del server | Stringa | "serverID" | X | X |
| logging 🡪 level | Il livello di gravità da loggare | INFO/WARNING/ CRITICAL/DEBUG | "INFO" | X | X |
| logging 🡪 path | Il percorso nel quale salvare il file di log | Percorso di sistema | “.” | X | X |
| logoPath | Il percorso nel quale trovare il file del logo | Percorso di sistema | "../logo.ico" | X | X |
| username | Il nome dell’utente che gestisce il centro di controllo | Stringa | "root" |  | X |
| password | La password dell’utente che gestisce il centro di controllo (salvata in SHA256) | Stringa | "4813494D137E1631BBA301D5ACAB6E7BB7AA74CE1185D456565EF51D737677B2"  (“root” in SHA256) |  | X |
| controlCenter 🡪 ip | L’indirizzo sul quale contattare il centro di controllo | IPv4 | "127.0.0.1" |  | X |
| controlCenter 🡪 port | La porta sulla quale contattare il centro di controllo | Numero intero positivo | 20211 |  | X |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| controlCenter 🡪 previewPort | La porta sulla quale trovare l’anteprima della notifica (l’indirizzo è uguale a quello del centro di controllo) | Numero intero positivo | 20212 |  | X |
| gui 🡪 ip | L’indirizzo sul quale lavora l’interfaccia grafica | IPv4 | "127.0.0.1" |  | X |
| gui 🡪 port | La porta sulla quale lavora l’interfaccia grafica | Numero intero positivo | 20213 |  | X |
| gui 🡪 templatesPath | Il percorso con i file HTML dell’interfaccia grafica | Percorso di sistema | "./gui/templates" |  | X |
| gui 🡪 resPath | Il percorso nel quale trovare le risorse CSS e JS dell’interfaccia grafica | Percorso di sistema | "./gui/static" |  | X |
| gui 🡪 sections | La lista di sezioni da caricare | Lista di stringhe | [“manager, “preview”, “list”] |  | X |
| bufferTimer | Il tempo di attesa tra i controlli del buffer, in secondi | Numero intero positivo | 30 |  | X |
| database 🡪 ip | L’indirizzo sul quale contattare il database | IPv4 | "127.0.0.1" |  | X |
| database 🡪 port | La porta sulla quale contattare il database | Numero intero positivo | 27017 |  | X |
| imagesPath | Il percorso con le immagini disponibili | Percorso di sistema | "./images" |  | X |
| scriptsPath | Il percorso con gli scripts disponibili | Percorso di sistema | "./scripts" |  | X |
| tempPath | Il percorso nel quale salvare gli script ricevuti | Percorso di sistema | "./temp" | X |  |
| website | L’URL del sito di riferimento dell’applicativo | Web URL | "http://gitsam.cpt.local/  2020\_2021\_2\_  semestre/butler" | X |  |
| modules 🡪 notification | Lo stato del modulo per le notifiche | Valore booleano | True | X |  |
| modules 🡪 inventory | Lo stato del modulo dell’inventario | Valore booleano | True | X |  |
| modules 🡪 inventory | Lo stato del modulo di comportamento | Valore booleano | True | X |  |
| automaticSendInterval | Il tempo tra l’invio spontaneo dei dettagli | Numero intero positivo | 30 | X |  |

# Avvio manuale

Per avviare i programmi manualmente sono disponibili il file **butler.py per il client**, e **manager.py per il server**. Eseguire qualunque altra classe ritorna degli errori.

# Uso del programma client

Una volta avviato automaticamente, non è più necessario interagirvi: un’icona nella barra delle applicazioni permetterà di vedere alcune informazioni ed eseguire alcune azioni, in nessun modo dannose per il server.

# Uso del programma server

Avviandolo con l’opzione --help è possibile vedere alcune modalità di avvio. In particolare, è possibile usarlo sia attraverso un’apposita interfaccia grafica (strutturata a sezioni, che si può scegliere di avviare o meno), sia attraverso richieste http/https con strumenti come cURL. Il primo metodo è consigliato per gli amministratori in quanto comodo e intuitivo, mentre il secondo potrebbe essere usato tramite degli scripts per eseguire operazioni in modo automatico: un caso d’uso reale potrebbe essere l’aggiunta ad un buffer della notifica che un aggiornamento è stato applicato durante la notte. L’accesso alla gestione del programma avviene attraverso le credenziali specificate nel file **config.json** (di default: **root** e **root**), nel quale la password deve essere salvata in hash SHA256. È quindi importante proteggere adeguatamente questo file.

## Nota

Il software è testato contro casi di uso normali, tipici di un ambiente di lavoro. Dato che non è da distribuire a qualunque utente ma rimane confinato a poche persone, non sono stati previsti particolari controlli (in particolare per quanto riguarda la GUI). Tutte le verifiche vengono effettuate lato server; se qualcosa non dovesse funzionare poiché i dati sono stati manomessi, il server si limiterà a ritornare un messaggio di errore oppure non eseguire l’operazione. In breve, non sono previste informazioni dettagliate sui possibili errori che esulano dall’uso comune.