Le interfacce di MiData

Sommario

Per ottenere dati da MiData esistono in linea di massima due possibilità:

Token di servizio

Gli account di servizio permettono di creare per un'applicazione esterna un account proprio con determinate autorizzazioni con il quale questa potrà quindi usare l'interfaccia JSON. Gli account di servizio vengono creati per un livello da una persona legittimata e rimangono anche quando questa persona lascia il gruppo o viene cancellata.

Specificazione (inglese)

OAuth

MiData è pure un OAuth 2.0 provider, ciò significa che un'applicazione esterna può autenticare gli utenti tramite MiData. Forse questo principio ti è già noto se hai già utilizzato "Registrarsi con Facebook" oppure "Registrarsi con Google". L'applicazione esterna può quindi ottenere informazioni relative all'utente da MiData se l'utente ne dà personalmente l'autorizzazione. La registrazione OAuth permette inoltre che l'applicazione esterna possa utilizzare l'interfaccia JSON a nome dell'utente. L'applicazione dispone delle stesse legittimazioni come l'utente.

Specificazione (inglese)

Se desideri utilizzare MiData come OAuth provider per una delle tue applicazioni, devi compilare la richiesta OAuth. L'assistenza IT del MSS ti assegnerà un accesso al sistema produttivo a condizione che tutte le condizioni preliminari siano soddisfatte: Richiesta Applicazione OAuth

Attualità: Il 14.09.2021 i token utenti personali sono stati segnalati come deprecati (deprecated) e prossimamente non saranno quindi più disponibili. Le applicazioni gestite finora con un token utente personale dovranno ora essere collegate tramite OAuth oppure i token di servizio.

Esempi di codice

Token di servizio, esempio con R

```r
# Install the required packages as follows:
# install.packages(c("jsonlite", "httr"))

library(jsonlite)
library(httr)

# Script parameters
baseurl <- "https://pbs.puzzle.ch"
token <- "d-Zx8kn-mKWaxXziYs62xVX2HdUdVKnSmLQYpQG-XznkbRD71g"
groupid <- 1

# Get MiData root group
rootElement <- fromJSON(paste(baseurl,"/groups/", groupid, ".json?token=", token, sep=""))
# Filter cantons
cantons = subset(rootElement$linked$groups, group_type == "Kantonalverband")
rownames(cantons) <- seq(length=nrow(cantons))

# Set a flag if the canton has an even id (just for fun)
cantons$even <- F
counter <- 1
for (id in cantons$id) {
       cantons$even[counter] <- ((as.numeric(cantons$id[counter]) %% 2) == 0)
       counter = counter + 1
}

# Print table of cantons
cantons
```

OAuth – Risorse utili

- Hitobito OAuth Flow
- Ruby (Doorkeeper, Devise, Omniauth) Example
- https://oauthdebugger.com
- https://www.oauth.com/playground/