Flickr zu Immich Migration

Vollständige Anleitung mit Docker/Podman

Plattformübergreifend: Linux, Mac, Windows

Plattform	Browser	Unterstützung
Linux	X11-Forwarding (automatisch)	Voll unterstützt
Mac	URL manuell öffnen	Unterstützt
Windows (WSL2)	URL manuell öffnen	Unterstützt

1. Übersicht

Diese Anleitung beschreibt die Migration einer Flickr-Fotobibliothek zu Immich unter Verwendung der Flickr-API. Der Prozess läuft in einem Docker/Podman-Container und funktioniert auf Linux, Mac und Windows.

Workflow:

- **flickr_download** - Fotos via API herunterladen
- **ExifTool** - Metadaten in Dateien einbetten
- **Immich CLI** - Upload zu Immich

Hinweis: Das Script erkennt automatisch das Betriebssystem und passt sich an.

2. Voraussetzungen

Komponente	Linux	Mac	Windows
Container Runtime	Docker oder Podman	Docker Desktop	Docker Desktop / WSL2
X11	Ja (xauth)	Nicht nötig	Nicht nötig
Shell	Bash (nativ)	Bash (nativ)	WSL2 oder Git Bash

3. Flickr API-Key erstellen

1. Gehe zu: https://www.flickr.com/services/apps/create/
2. Klicke auf 'Request an API Key' → 'Non-Commercial'
3. Fülle das Formular aus (Name: z.B. 'Flickr Backup')
4. Notiere **API Key** und **API Secret**

4. Script einrichten

Download und Ausführbar machen:

```
# Script herunterladen (oder aus Anhang kopieren)
chmod +x flickr-docker.sh

# System-Info anzeigen
./flickr-docker.sh info
```

Docker-Image bauen:

```
./flickr-docker.sh build
```

API-Konfiguration erstellen:

```
mkdir -p flickr-config
cat > flickr-config/.flickr_download << EOF
api_key: DEIN_API_KEY
api_secret: DEIN_API_SECRET
EOF
```

5. Authentifizierung

Linux:

Unter Linux öffnet sich automatisch ein Browser-Fenster für die OAuth-Authentifizierung.

```
# Standard (Chrome)
./flickr-docker.sh auth

# Mit Firefox
BROWSER=firefox ./flickr-docker.sh auth
```

Mac / Windows:

Auf Mac und Windows wird die OAuth-URL im Terminal angezeigt. Diese URL muss manuell im Browser geöffnet werden.

```
./flickr-docker.sh auth

# Ausgabe:
# HINWEIS für Mac/Windows:
#   - Eine URL wird im Terminal angezeigt
#   - Öffne diese URL manuell in deinem Browser
#   - Nach dem Login wird der Callback automatisch verarbeitet
```

Wichtig: Nach erfolgreichem Login wird das Token in flickr-config/.flickr_token gespeichert.

6. Fotos herunterladen

Alle Alben eines Benutzers:

```
# Alle Alben herunterladen
./flickr-docker.sh download dein_flickr_username

# Oder mit voller URL
./flickr-docker.sh download https://www.flickr.com/photos/username/
```

Alben auflisten:

```
./flickr-docker.sh list dein_flickr_username
```

Einzelnes Album:

```
./flickr-docker.sh album 72157622764287329
```

Die Fotos werden in **./flickr-backup/** gespeichert, zusammen mit JSON-Metadaten.

7. Metadaten einbetten

Nach dem Download müssen die Flickr-Metadaten (Datum, Titel, Tags) in die Bilddateien eingebettet werden, damit Immich sie korrekt erkennt.

Script für Metadaten-Embedding:

```
#!/bin/bash
# embed_metadata.sh

BACKUP_DIR="./flickr-backup"

find "$BACKUP_DIR" -name "*.json" | while read json_file; do
    base="${json_file%.json}"

    # Finde zugehörige Mediendatei
    for ext in jpg jpeg png gif mp4 mov avi; do
        media_file="${base}.${ext}"
        [ -f "$media_file" ] && break
    done

    [ ! -f "$media_file" ] && continue

    # Datum extrahieren und einbetten
    date_taken=$(jq -r ".datetaken" "$json_file")
    if [ "$date_taken" != "null" ]; then
        exif_date=$(echo "$date_taken" | sed "s/-/:/g")
        exiftool -overwrite_original \
            -DateTimeOriginal="$exif_date" \
            -CreateDate="$exif_date" \
            "$media_file"
    fi
done
```

Hinweis: Videos benötigen andere EXIF-Tags (MediaCreateDate, TrackCreateDate).

8. Upload zu Immich

Immich CLI installieren:

```
npm install -g @immich/cli
```

Login:

```
immich login https://immich.example.com API_KEY
```

Upload:

```
# Alle Fotos hochladen
immich upload ./flickr-backup --recursive

# Mit Album-Erstellung
immich upload ./flickr-backup --recursive --album
```

9. Befehlsreferenz

Befehl	Beschreibung
build	Docker-Image bauen
auth	OAuth-Authentifizierung starten
download <user>	Alle Alben eines Users herunterladen
album <id>	Einzelnes Album herunterladen
list <user>	Alben eines Users auflisten
shell	Interaktive Shell im Container
test-browser [url]	X11-Verbindung testen (nur Linux)
info	System-Informationen anzeigen
clean	Image und temp. Dateien löschen

Umgebungsvariablen:

Variable	Beschreibung	Default
BROWSER	Browser für OAuth (Linux)	chrome
BROWSER	Browser für OAuth (Mac/Win)	<leer> = System

10. Fehlerbehebung

Browser öffnet nicht (Linux):

```
# DISPLAY prüfen
echo $DISPLAY

# xauth prüfen
xauth list

# Test
./flickr-docker.sh test-browser
```

Token ungültig:

```
# Token löschen und neu authentifizieren
rm flickr-config/.flickr_token
./flickr-docker.sh auth
```

Podman-Probleme:

```
# Das Script erkennt Podman automatisch und verwendet:
```

```
# --userns=keep-id (für X11-Zugriff)
# --security-opt label=disable (für SELinux)
```

Mac/Windows OAuth-Callback:

Falls der OAuth-Callback nicht funktioniert, prüfe ob die Ports 8080-8100 frei sind. Das Script published diese Ports automatisch für den Callback-Server.

11. Verzeichnisstruktur

Verzeichnis	Inhalt
./flickr-config/	API-Keys (.flickr_download) und Token (.flickr_token)
./flickr-backup/	Heruntergeladene Fotos und JSON-Metadaten
./flickr-cache/	API-Cache für Resume-Funktion