# Importer

- MoWaS - Modulares Warnsystem

# MoWaS - Modulares Warnsystem

Anbindung an das System von Bund und Ländern für Warnungen und Gefahreninformationen.

## Funktionsprinzip

MoWaS übermittelt die Meldung als XML Datei und läd diese auf den von uns zur Verfügung gestellten SFTP Zugang hoch. Der "Watcher" überwacht das Import Verzeichis und erstellt ein Job sowie eine neue Datei eintrifft. Der "Worker" verarbeiet die XML und schickt die Daten an den entsprechenden SVA JSON Server.

Das Mapping zum entsprechenden JSON Server erfolgt über den *Regionalschlüssel* auf **Kreis Ebene** der Meldung. Details zum Aufbau des Regionalschlüssels wird hier ausführlich erklärt: verwaltungsgliederung_vg.pdf und den passenden Regionalschlüssel findet man in der struktur_und_attribute_vg250.xls im Reiter **VG250_Punkt**

## Konfiguration

Die Konfiguration befindet sich als statische JSON Datei im Git Repository https://gitlab.tpwd.de/ikusei/smart-village-app-mowas in der `config/locations.json`.

### Neue Standorte

Infos zu den Orten und deren RS (Regionalschlüssen ) befinden sich in der struktur_und_attribute_vg250.xls im Reiter: **VG250_Punkt**

Für einen neuen Eintrag einer Kommune werden folgende Informationen benötigt.

1. Name der Kommune
2. Regionalschlüssel
3. API Endpunkt vom JSON Server der Kommune
4. Auth Endpunt vom Main Server der Kommune
5. OAuth Client ID und Secret

Beispiel:

```
{
  "name": "Internal Development",
  "rs": "999999999999",
  "endpoint": {
    "json_url": "https://json.int-development.smart-village.app/api",
    "auth_url": "https://server.int-development.smart-village.app/oauth/token",
    "client_id": "mOBa42XCe_yl3_u_ipUka8tc2oR6TMX8GATzVeZMNek",
    "client_secret": "Zb7iseniNw2eFtrmjQgmUuwbX03Xvimdps6D2c6nIS8"
  }
}
```


# Testsendungen

In unbestimmten Abständen werden Testsendungen seitens MoWaS durchgeführt. Diese werden mit dem **Status** `Test` gekennzeichnet. Diese Meldungen werden nicht an den JSON Server gesendet. Es findet eine Benachrichtigung per E-Mail und im Slack Channel **#z-log-smartvillage-app** statt.

# Dashboard

Über das Dashboard können alle eingegangenen Meldungen und deren Verarbeitung eingesehen werden.
Erreichbar unter für die jeweilige Umgebung unter:

- https://mowas-dashboard.sva.customer.planetary-quantum.net
- https://mowas-staging-dashboard.sva.customer.planetary-quantum.net

# Production

## SFTP Zugang

Der Zugang kann über einen beliebigen FTP Client erfolgen oder über den SFTPGo Web Client. Die Zugangsdaten sind im 1Password hinterlegt.

**SFTPGo Web Client**

https://mowas-ftp.sva.customer.planetary-quantum.net/web/client

**FTP Client**

Der Kundenzugang läuft über den User `mecom` passwortlos mittels SSH Key.

```
Host: mowas.smart-village.app
Port: 2222
User: tpwd
Pass: <1Password>
```

# Staging

**SFTPGo Web Client**

https://mowas-staging-ftp.sva.customer.planetary-quantum.net/web/client

**FTP Client**

Kundenzugang zum Staging ist derzeit nicht vorgesehen.

```
Host: mowas.smart-village.app
Port: 8888
User: tpwd
Pass: <1Password>
```

# Besonderheiten der Staging Umgebung

Das Funktionsprinzip ist identisch zur Production Umgebung. Die Verarbeitung läuft genauso, mit dem Unterschied das die Daten am Ende immer and "Internal Development" JSON Server mit dem *Regionalschlüssel* 999999999999 gesendet werden. Eine Benachrichtigung per E-Mail und im Slack Channel für Testsendungen findet ebenfalls statt.

**Dashboard**
Über das Dashboard können alle eingeganenen Meldungen und deren Verarbeitung eingesehen werden. Erreichbar unter: https://mowas-staging-dashboard.sva.customer.planetary-quantum.net

# Stack

sftpgo
sidekiq / redis
watcher (listen)

worker (sidekiq)

# Development

1.) redis-server
2.) ruby bin/watcher.rb
3.) sidekiq -r ./bin/worker.rb
4.) statt ftp benutzt man das Importverzeichnis `import/new/`

# Debugger

```
bundle exec bin/debugger.rb import/processed/
```