SituationReport

get_data

Manual de Utilizator

Exportați date Jira pentru SituationReport

---

situation-report -- github.com/Jaegerfeld/situation-report

Pentru Agile Coaches si Manageri PI -- Version 0.19.0

1 Exportul datelor din Jira Cloud

> Notă: Modulul get_data este încă în curs de dezvoltare. Acest manual descrie exportul manual al datelor Jira prin intermediul API-ului REST Jira. Acesta este același proces pe care get_data îl va automatiza — fișierele JSON exportate pot fi procesate direct de transform_data.

Acest capitol explică cum să exportați date reale despre issue-uri din Jira Cloud pentru SituationReport. Exportul produce aceeași structură JSON ca și Generatorul de date de test — transform_data le procesează pe ambele identic.

1.1 Ce aveți nevoie

Cerință	Detalii
Cont Jira Cloud	Un cont cu acces de citire la proiectul dorit
Token API	Token personal de la id.atlassian.com — creat o singură dată
Cheia proiectului	Codul scurt al proiectului Jira, ex. 'ART_A' sau 'SCRUM'
curl sau browser	curl pentru interogări automate; browser pentru teste rapide

1.2 Crearea unui token API

Token-ul API înlocuiește parola pentru apelurile API și se creează o singură dată:

Pas	Acțiune
1	Deschideți https://id.atlassian.com în browser (conectat la Jira)
2	Security → API tokens → Create API token
3	Introduceți o etichetă, ex. 'SituationReport', faceți clic pe 'Create'
4	Copiați token-ul și păstrați-l în siguranță — este afișat o singură dată!

Notă de securitate: Tratați token-ul API ca pe o parolă. Păstrați-l într-un manager de parole și nu îl distribuiți.

1.3 Interogarea API-ului REST Jira

Jira Cloud oferă două variante de API. Utilizați API-ul v3 curent pentru scripturile noi; scripturile v2 existente continuă să funcționeze. `expand=changelog` este obligatoriu în ambele variante — fără el, tranzițiile de status lipsesc și transform_data nu poate calcula valorile de timp pe etapă.

API curent (v3) – recomandat

Endpoint: `POST /rest/api/3/search/jql`

Parametrii sunt transmiși ca corp JSON al cererii. Răspunsul include suplimentar `isLast` și `nextPageToken` pentru paginare (Secțiunea 1.5).

```
curl -X POST \
  "https://company.atlassian.net/rest/api/3/search/jql" \
```

```
  -u "name@company.com:YourAPIToken" \
  -H "Content-Type: application/json" \
  -d '{"jql":"project=ART_A ORDER BY created ASC",
       "maxResults":100,
       "expand":["changelog"],
       "fields":["issuetype","created",
                 "status","summary"]}' \
  -o ART_A_page1.json
```

maxResults este limitat la 100 de issue-uri per cerere cu API-ul v3. Utilizați nextPageToken pentru proiecte cu mai mult de 100 de issue-uri (Secțiunea 1.5).

API mai vechi (v2) – încă suportat

Endpoint: `GET /rest/api/2/search`

Parametrii ca șir de interogare URL. Suportă până la 1.000 de issue-uri per cerere și paginare bazată pe offset cu `startAt`.

```
curl -u "name@company.com:YourAPIToken" \
  "https://company.atlassian.net/rest/api/2/search?\
jql=project=ART_A+ORDER+BY+created+ASC\
&expand=changelog\
&maxResults=1000\
&fields=issuetype,created,status,project,summary,resolution" \
  -o ART_A_page1.json
```

Sfat: Pentru scripturile noi, API-ul v3 este preferat. Scripturile v2 existente nu trebuie migrate.

1.4 Câmpurile necesare pentru transform_data

Câmp Jira	Obligatoriu	Scop
key	□	Identificatorul issue-ului (ex. ART_A-001)
fields.issuetype.name	□	Tipul issue-ului pentru filtrare (Feature, Bug, ...)
fields.created	□	Data creării issue-ului
fields.status.name	□	Statusul curent
changelog (expand=changelog)	□	Tranziții de status cu marcaje de timp — TREBUIE setat în URL
fields.resolution	–	Opțional — tipul rezoluției (ex. 'Done', 'Won't Fix')
fields.summary	–	Opțional — titlul issue-ului

1.5 Paginare – mai mult de 100 de issue-uri

API v3 – Paginare bazată pe cursor (nextPageToken)

API-ul v3 returnează cel mult 100 de issue-uri per cerere. Fiecare răspuns conține fie un `nextPageToken` pentru pagina următoare, fie `"isLast": true` când toate issue-urile au fost preluate. Paginile trebuie preluate secvențial — token-ul din răspunsul curent se folosește în cererea următoare.

```
# Pagina 1 – fara nextPageToken
curl -X POST \
  "https://company.atlassian.net/rest/api/3/search/jql" \
  -u "name@company.com:Token" \
  -H "Content-Type: application/json" \
  -d '{"jql":"project=ART_A ORDER BY created ASC",
       "maxResults":100,"expand":["changelog"],
       "fields":["issuetype","created",
                 "status","summary"]}' \
  -o ART_A_page1.json

# Cititi nextPageToken din pagina 1:
#   cat ART_A_page1.json | grep nextPageToken
#   (sau cu jq: jq -r '.nextPageToken' ART_A_page1.json)

# Pagina 2 – inserati nextPageToken
curl -X POST \
  "https://company.atlassian.net/rest/api/3/search/jql" \
  -u "name@company.com:Token" \
  -H "Content-Type: application/json" \
  -d '{"jql":"project=ART_A ORDER BY created ASC",
       "maxResults":100,"expand":["changelog"],
       "fields":["issuetype","created",
                 "status","summary"],
       "nextPageToken":"<token din pagina 1>"}' \
  -o ART_A_page2.json

# Repetati pana cand isLast:true apare in raspuns

# Imbinati toate paginile
python -m helper ART_A_page1.json ART_A_page2.json ... \
  --output ART_A_merged.json
```

Repetați până când răspunsul conține `"isLast": true`. Helper îmbină toate paginile și elimină duplicatele automat.

API v2 – Paginare bazată pe offset (startAt) – Moștenit

API-ul v2 suportă până la 1.000 de issue-uri per cerere. Cu `startAt`, orice pagină poate fi accesată direct. Important: `startAt` este compatibil doar cu endpoint-ul v2 — nu cu noul endpoint v3.

```
# Pagina 1 (issue-urile 1-1.000)
curl -u "name@company.com:Token" \
  "https://company.atlassian.net/rest/api/2/search?\
```

```
jql=project=ART_A&expand=changelog&maxResults=1000&startAt=0" \
  -o ART_A_page1.json

# Pagina 2 (issue-urile 1.001-2.000)
curl -u "name@company.com:Token" \
  "https://company.atlassian.net/rest/api/2/search?\
jql=project=ART_A&expand=changelog&maxResults=1000&startAt=1000" \
  -o ART_A_page2.json

# Imbinati fisierele cu Helper
python -m helper ART_A_page1.json ART_A_page2.json \
  --output ART_A_merged.json
```

Incrementați `startAt` cu 1.000 până când răspunsul conține mai puțin de 1.000 de issue-uri.

1.6 De la exportul JSON la fișierul de flux

După export, creați un fișier de flux care mapează statusurile reale din Jira. Următorul fragment Python extrage toate numele de statusuri din export:

```
import json
data = json.loads(open('ART_A_page1.json').read())
statuses = set()
for issue in data['issues']:
    for h in issue['changelog']['histories']:
        for item in h['items']:
            if item['field'] == 'status':
                statuses.add(item['toString'])
print(sorted(statuses))
```

Din numele de statusuri găsite, creați un `workflow.txt` — ordonați statusurile în ordine logică și setați marcatorii `<First>` și `<Closed>`. Formatul fișierului de flux este descris în Manualul de Utilizator transform_data.

2 Întrebări frecvente (FAQ)

Ce se întâmplă dacă expand=changelog lipsește?

transform_data nu găsește nicio tranziție de status și produce valori de timp goale pentru toate issue-urile. Fișierele Excel nu conțin date utilizabile. Includeți întotdeauna `expand=changelog` în URL.

Primesc o eroare 'Unauthorized' la apelul curl.

Verificați: Adresa de e-mail este corectă? Token-ul API este corect? Sunt separate prin două puncte? Contul dvs. are acces de citire la proiect? Sfat: Generați un nou token API dacă cel vechi a fost pierdut sau a expirat.

Pot îmbina mai multe proiecte Jira?

Da — fie prin JQL (`jql=project in (ART_A, ART_B)`) pentru un singur export, fie îmbinând două exporturi separate cu Helper. Asigurați-vă că fluxurile de lucru ale ambelor proiecte sunt compatibile.

Cât de mare poate deveni fișierul JSON?

Regulă generală: aproximativ 2–5 KB per issue (în funcție de lungimea changelog-ului). 1.000 de issue-uri ≈ 2–5 MB. Pentru proiecte foarte mari (10.000+ issue-uri), utilizați paginarea și Helper-ul.

Ar trebui să folosesc date de test sau date reale?

Pentru demonstrații, instruire și teste de instalare, datele sintetice de test din Testdata Generator sunt alegerea mai bună — fără probleme de confidențialitate, reproductibile și complet controlate. Pentru analize reale de flux și decizii, datele reale Jira reflectă starea reală a fluxului de lucru.

3 Glosar

Termen	Explicație
ART	Agile Release Train — o structură de echipe în SAFe
API	Application Programming Interface — interfață de programare
Token API	Cheie de securitate pentru accesul la API-ul REST Jira; înlocuiește parola în apelurile curl
Changelog	Jurnalul Jira al tuturor modificărilor de status pentru un issue (necesită expand=changelog)
CFD	Cumulative Flow Diagram — arată numărul cumulativ de issue-uri per etapă în timp
Etapa Closed	Etapa care marchează un issue ca finalizat (workflow.txt: <Closed>)
Timp de ciclu	Timp de la prima etapă activă (<First>) până la etapa closed
Etapa First	Prima etapă activă de unde începe măsurarea timpului de ciclu (workflow.txt: <First>)
Helper	Modulul SituationReport pentru îmbinarea mai multor fișiere JSON Jira
Issue	Un element de lucru în Jira (Feature, Bug, Enabler, Story, …)
JQL	Jira Query Language — limbaj de interogare pentru căutări Jira, ex. project=ART_A
JSON	JavaScript Object Notation — formatul exporturilor API Jira
Etapă	Un pas în fluxul de lucru (corespunde unei coloane sau unui nume de status Jira)
Flux de lucru	Secvența ordonată de etape prin care trece un issue