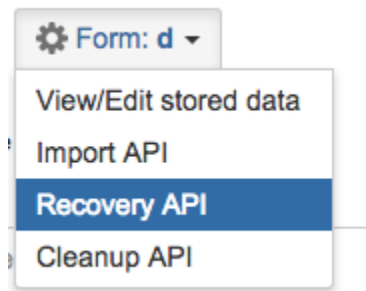


Migration and Recovery API

Generally available for ConfiForms form administrators with **Confluence admin permissions** and for users who are **Confluence space admins of the space where this form is located**

- [Manual process](#)
- [Automation](#)
- [How to run/execute this](#)

Manual process



You are presented with a screen similar to this

⚠ This operation will replace the data stored by ConfiForms with the data in the uploaded file. You need to provide the file in a RAW format.
This operation is not recoverable. This means, you will not be able to access the data you had before the recovery operation

ConfiForms Form "confiFormsIssues" dataset RAW export: [Download RAW](#)

Page: [ConfiForms Backlog \(3670127\)](#)

Page ID *

3670127

Type page numeric ID

Form name *

confiFormsIssues

XML/RAW JSON file *

Choose file No file chosen

ConfiForms XML file in internal format (or exported RAW JSON format)

Upload the file and load the data

Uploading the data into the form will replace existing data. No IFTTT rules will be executed

Use case on Recovery API - [How to copy the data from one form to another with the same structure](#)

Automation

We have developed a simple Java based script that downloads the data in a RAW format from the source server/page and attempts to upload it to the target server



conforms-migration.zip

Script is pretty simple and all you have to do is to set the following properties

```
// source server details and source form name and location
static String sourceServer = "https://wiki.vertuna.com/";
static String sourceFormName = "form1";
static long sourcePageId = 64063006;
static String sourceServerUsername = "";
static String sourceServerPassword = "";

// target server details and target form name and location
static String targetServer = "http://localhost:1990/confluence/";
static String targetFormName = "form1";
static long targetPageId = 786437;
static String targetServerUsername = "";
static String targetServerPassword = "";
```

Full listing of the script (same as in the zip file linked earlier)

```
package com.vertuna.plugins.confluence.migration;

import org.apache.commons.codec.binary.Base64;
import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.HttpMethod;
import org.apache.commons.httpclient.URI;
import org.apache.commons.httpclient.UsernamePasswordCredentials;
import org.apache.commons.httpclient.auth.AuthScope;
import org.apache.commons.httpclient.methods.GetMethod;
import org.apache.commons.httpclient.methods.PostMethod;
import org.apache.commons.httpclient.methods.multipart.FilePart;
import org.apache.commons.httpclient.methods.multipart.MultipartRequestEntity;
import org.apache.commons.httpclient.methods.multipart.Part;
import org.apache.commons.httpclient.methods.multipart.StringPart;
import org.apache.commons.io.FileUtils;

import java.io.File;
import java.io.IOException;
import java.util.logging.Level;

public class MigrateFormData {
    // source server details and source form name and location
    static String sourceServer = "https://wiki.vertuna.com/";
    static String sourceFormName = "form1";
    static long sourcePageId = 64063006;
    static String sourceServerUsername = "";
    static String sourceServerPassword = "";

    // target server details and target form name and location
    static String targetServer = "http://localhost:1990/confluence/";
```

```

static String targetFormName = "form1";
static long targetPageId = 786437;
static String targetServerUsername = "";
static String targetServerPassword = "";

public static void main(String[] args) throws IOException {
    // no logging from http client
    java.util.logging.Logger.getLogger("org.apache.commons").setLevel(Level.OFF);

    File confiformsExportFile = File.createTempFile("confiforms", "");
    confiformsExportFile.deleteOnExit();

    try {
        String sourceUrl = sourceServer + "/ajax/confiforms/rest/raw.action?pageId=" + sourcePageId + "&fd=" +
sourceFormName + ":" + sourcePageId;
        System.out.println("Requesting source: " + sourceUrl);
        byte[] rawExportContents = makeRequest(sourceUrl, sourceServerUsername, sourceServerPassword, url -> new
GetMethod(url));
        FileUtils.writeByteArrayToFile(confiformsExportFile, rawExportContents);

        String targetUrl = targetServer + "/ajax/confiforms/rest/raw.action";
        System.out.println("Uploading to: " + targetUrl);

        makeRequest(targetUrl, targetServerUsername, targetServerPassword, url -> {
            Part[] parts = new Part[4];
            parts[0] = (new StringPart("pageId", String.valueOf(targetPageId)));
            parts[1] = (new StringPart("subAction", "replace"));
            parts[2] = (new StringPart("fd", targetFormName + ":" + targetPageId));
            parts[3] = (new FilePart("rawFile", confiformsExportFile));

            PostMethod m = new PostMethod(url);
            m.setRequestEntity(new MultipartRequestEntity(parts, m.getParams()));
            return m;
        });
    } finally {
        confiformsExportFile.delete();
    }
    System.out.println("Done");
}

interface PrepareMethodCallback {
    HttpMethod prepare(String url) throws IOException;
}

public static byte[] makeRequest(String url, String username, String password, PrepareMethodCallback
prepareMethodCallback) throws IOException {
    HttpClient client = new HttpClient();
    client.getState().setAuthenticationPreemptive(true);
    client.getState().setCredentials(AuthScope.ANY, new UsernamePasswordCredentials(username, password));

    HttpMethod m = prepareMethodCallback.prepare(url);
    try {
        m.addRequestHeader("Accept", "application/json");

        String encodedAuth = new String(Base64.encodeBase64((username + ":" + password).getBytes()));
        m.addRequestHeader("Authorization", "Basic " + encodedAuth);
        m.setDoAuthentication(true);

        int responseCode = client.executeMethod(m);
        if (responseCode == 301 || responseCode == 302) {
            String location = m.getResponseHeader("Location").getValue();
            m.setURI(new URI(location, false));
            responseCode = client.executeMethod(m);
            if (responseCode == 200 || responseCode == 201 || responseCode == 202 || responseCode == 204) {
                return m.getResponseBody();
            }
        } else {
            if (responseCode == 200 || responseCode == 201 || responseCode == 202 || responseCode == 204) {
                return m.getResponseBody();
            }
        }
        throw new IOException("Requested to " + url + " has resulted in error. Response code = " + responseCode +

```

```
". Response contents: " + m.getResponseBodyAsString());
    }
    } finally {
        m.releaseConnection();
    }
    throw new IOException("Cannot complete request to " + url);
}
}
```

And the following dependencies are configured in the maven pom.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
<modelVersion>4.0.0</modelVersion>

<groupId>com.vertuna.plugins.confluence</groupId>
<artifactId>conforms-data-migration</artifactId>
<version>1.0</version>

<properties>
<maven.compiler.source>8</maven.compiler.source>
<maven.compiler.target>8</maven.compiler.target>
</properties>

<repositories>
<repository>
<id>atlassian-public</id>
<url>https://maven.atlassian.com/maven-external</url>
<snapshots>
<enabled>true</enabled>
<updatePolicy>never</updatePolicy>
<checksumPolicy>warn</checksumPolicy>
</snapshots>
<releases>
<enabled>true</enabled>
<checksumPolicy>warn</checksumPolicy>
</releases>
</repository>
<repository>
<id>maven-atlassian-com</id>
<name>Atlassian Public Repository</name>
<url>https://packages.atlassian.com/maven/public</url>
</repository>
</repositories>

<dependencies>
<dependency>
<groupId>commons-httpclient</groupId>
<artifactId>commons-httpclient</artifactId>
<version>3.1-atlassian-2</version>
</dependency>
<dependency>
<groupId>commons-lang</groupId>
<artifactId>commons-lang</artifactId>
<version>2.6</version>
</dependency>
<dependency>
<groupId>commons-io</groupId>
<artifactId>commons-io</artifactId>
<version>2.6</version>
</dependency>
</dependencies>

</project>

```

How to run/execute this

You will need Apache Maven (<https://maven.apache.org/download.cgi>) and Java installed on your computer, then navigating to the directory where the pom.xml file is located, via command prompt, you will need to type the following commands

```

mvn compile
mvn exec:java -Dexec.mainClass=com.vertuna.plugins.confluence.migration.MigrateFormData

```

