PUBLIC

SAP HANA Platform 2.0 SPS 00
Document Version: 1.0 – 2016-11-30

SAP File Processing for SAP HANA

Enabling Text and Metadata Extraction from Unstructured Content

Content

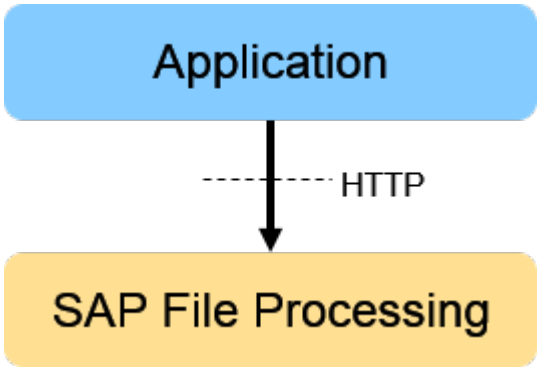
1 About This Guide

This guide is a detailed description of SAP File Processing including the concepts, programming APIs and operation. System architects will be able to integrate SAP File Processing in business processes and system landscapes. Administrators will learn how to set up and operate File Processing and developers will find detailed information on programming APIs.

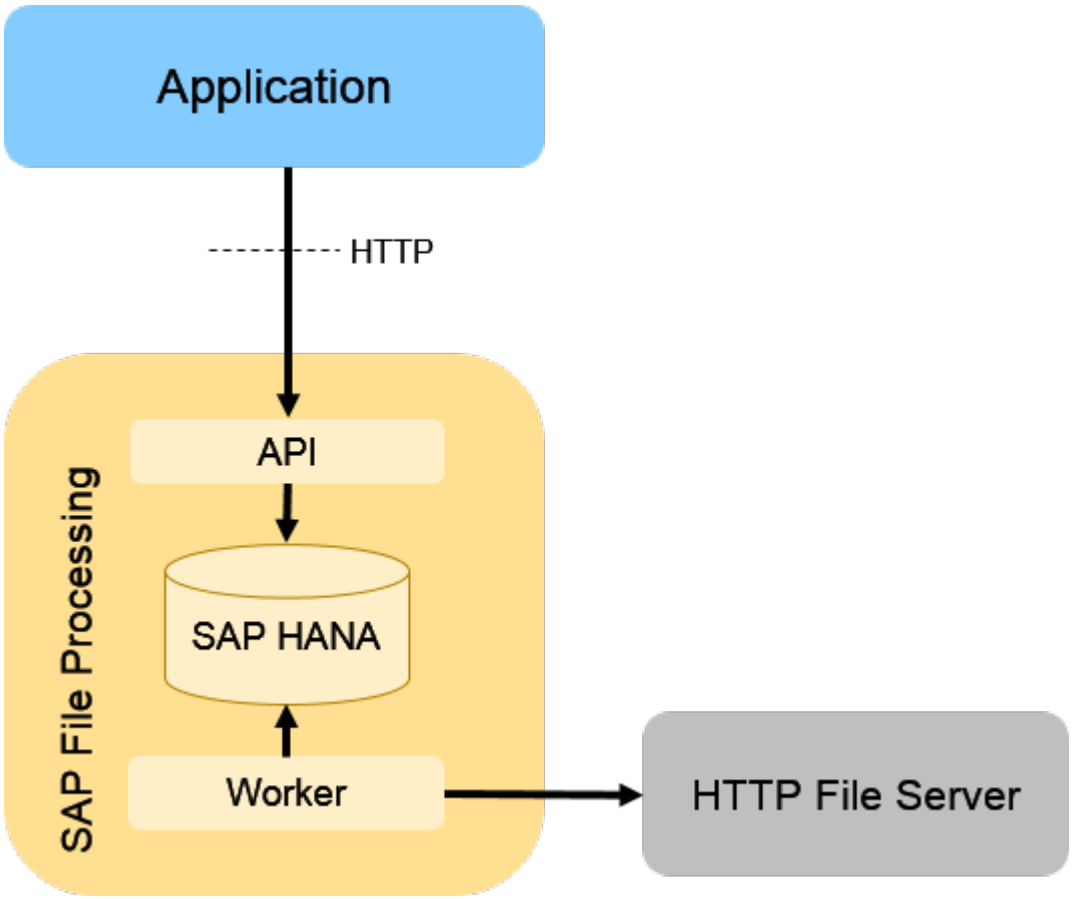
SAP File Processing is a component of SAP HANA that provides structured information from unstructured files. The rich set of HTTP APIs enables application programmers to integrate SAP File Processing features in client applications.

2 Concept and Architecture

SAP File Processing exposes an HTTP REST API that is consumed by an application. HTTP is the only channel to access the API.



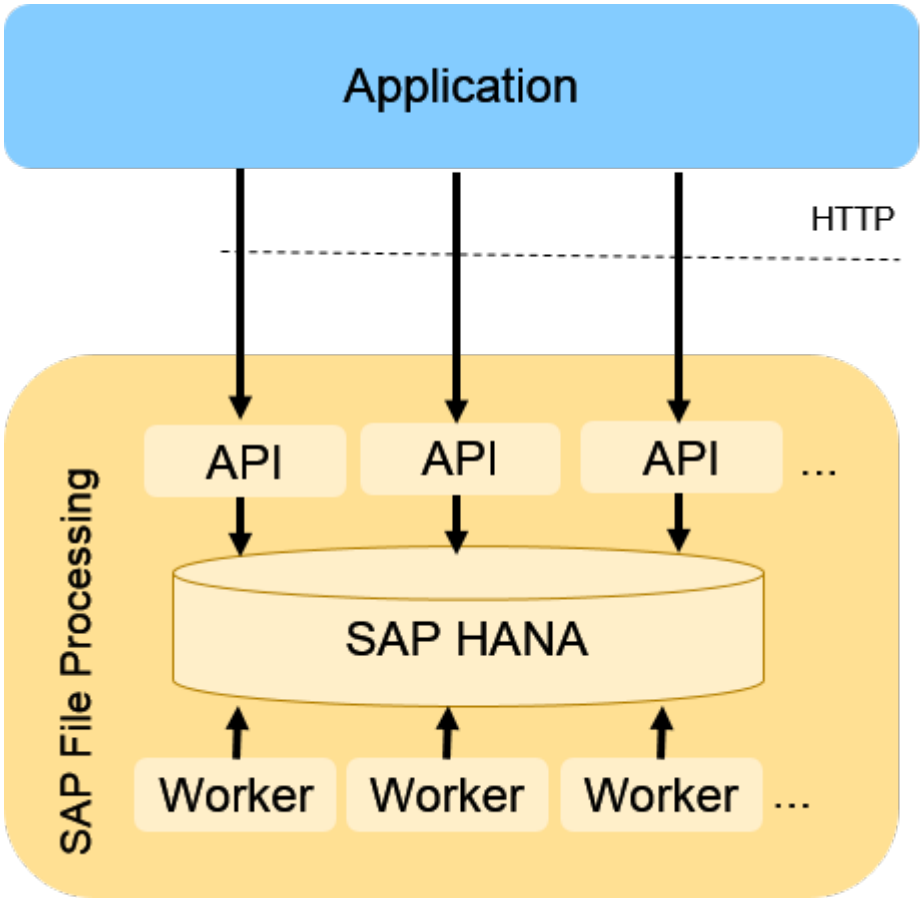
SAP File Processing has an API component to communicate with the application. The API uses the SAP HANA database to store file processing information. The worker component processes the files and uses the SAP HANA database to store results and uses HTTP file servers to acquire the file content.



Scalability

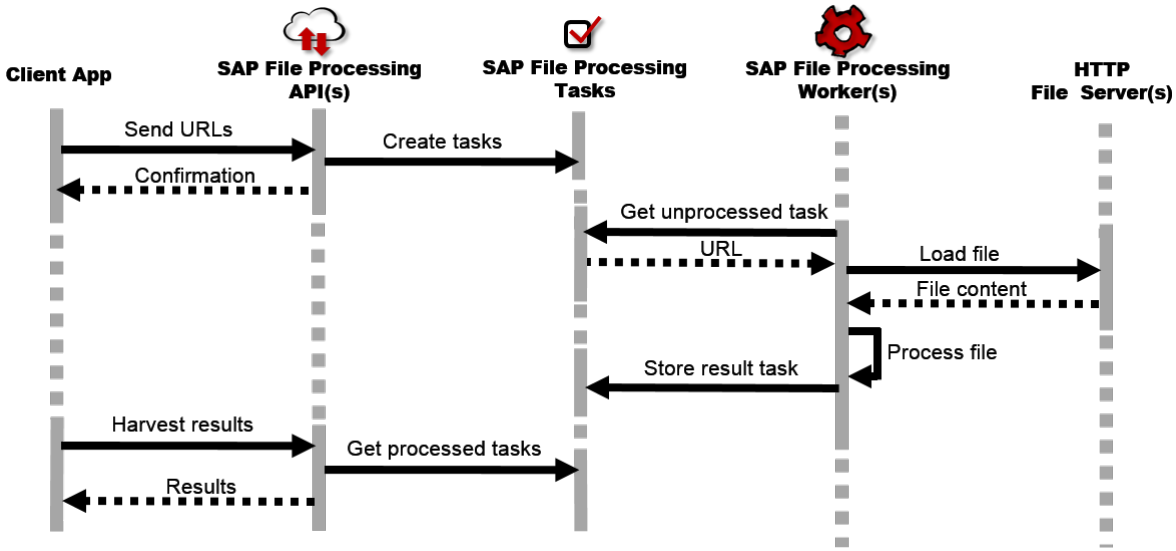
Components of SAP File Processing are scaled up and down to respond to varying demands. The API component is responsible to fulfill HTTP requests. Additional API instances will serve increasing API calls. One API component is started by default (the master application).

To increase the file processing speed, increase the number of worker components. Three workers start by default. The number of workers correlates with the number of parallel processed files.



The Central Process Flow

SAP File Processing is able to process large volumes of files. The files are processed asynchronously. A client application sends URLs to SAP File Processing for processing. The API confirms the request. SAP File Processing creates a SAP File Processing task for every file URL. The task describes the processing status of the file and is used to provide results or error messages in case of errors.



The SAP File Processing worker picks an unprocessed task and loads the content of the file. After the file is loaded, the plain text content is extracted from the original file and other metadata is determined. The results are stored in the task.

The client application harvests the results of the processed tasks. The application uses the harvested data within the application.

Main Terminology

Table 1:

Term	Description
File	The object that can be processed by SAP File Processing. E-Mails, office files, PDFs and many more types can be processed. A list of supported file types is provided as a service operation (`GET api/v4/mimetypes`).
Job	The job describes how the files have to be processed. The job is used to separate the processing from the client's perspective. Jobs are processed according their priority and can be limited to a certain processing power. A job can process many files.
Task	The task describes the current processing state of the file. A task has exactly one file. Tasks are used to monitor the file processing and serve as a temporary store for the file and the determined metadata.

Term	Description
Worker	The worker is the application instance that processes tasks. SAP File Processing can run multiple workers in parallel to support high scale processing.
API / Master	The API or the master application is the HTTP server interface of SAP File Processing.

3 Setting Up SAP File Processing

Prerequisites

- You are using SAP HANA 2.0, including XS Advanced.
- You downloaded the ZIP archive for *XSA File Processor 1.0* from the SAP Software Download Center:
  1. Go to https://support.sap.com/swdc.
  2. Choose *Software Downloads* > *Installation and Upgrades* > *Software Downloads* > *INSTALLATION AND UPGRADES* > *By Alphabetical Index (A-Z)*.
  3. Choose *H* and search for *SAP HANA Platform Edition 2.0*.
  4. On the product page select *Support packages and Patches*.
  5. On the *DOWNLOADS* tab, select *ENTRY BY COMPONENT* > *XSAC FILEPROCESSOR 1.0*.
  6. Download the ZIP archive.
  7. Unpack the ZIP archive and save the extracted files to a local folder.
  8. Copy the extracted files from the local folder to your SAP HANA system (using the `<sid>adm` user).

For more information, refer to the *SAP HANA Server Installation and Update Guide*.

Procedure

1. Use the SAP HANA Database Lifecycle Manager `hdblcm` to deploy SAP File Processing. The app will be deployed in the SAP space.

   Refer to the SAP HANA Lifecycle Management chapter in the *SAP HANA Administration Guide*.

   Note

   As the data of SAP File Processing is separated by deployed instances, it is necessary to provide a separate instance for every tenant. Deploy separate instances of SAP File Processing for every tenant you are using. Example: If SAP File Processing is connected to an SAP Business Suite system, every client requires a separate instance of SAP File Processing.

2. Verify that **SAP File Processing** is deployed successfully and is up and running:

   ```
   xs a
   ```

   Sample Code

   ```
   Getting apps in org "your org" / space "your space" as XSA_ADMIN...
           Found apps:
           name requested state instances memory disk urls
           fileprocessor-db STOPPED 0/1 256 MB <unlimited> <none>
   ```

```
        fileprocessor-master STARTED 1/1 1,00 GB <unlimited> https://server:
51057
        fileprocessor-web STARTED 1/1 1,00 GB <unlimited> https://server:
51056
        fileprocessor-worker STARTED 3/3 1,00 GB <unlimited> <none>
```

Four apps are listed. The `master`, `web` and `worker` apps should be started. The `db` app already stopped, as this app set up the database. The SAP File Processing application endpoint URL is the `fileprocessor-web` url (`https://server:51056` in this example).

SAP File Processing is now installed successfully. However, a user cannot access the application yet.

3. Determine the application name.

   The application name is required when the role collection is edited. To determine the application name, enter the command `xs env fileprocessor-web`.

   You find the application name in the section `VCAP_SERVICES-credentials-xsappname`.

Related Information

SAP HANA Adminstration Guide
SAP HANA Server Installation and Update Guide
SAP HANA Developer Guide For SAP HANA XS Advanced Model

3.1 Configuring Role Collection and Users

Procedure

1. Determine the XS Advanced admin web UI endpoint.

   Enter the command `xs version`.

   In the output, you find the **xsa-admin URL** under `Registered service URLs`. Use this URL to access the XS Advanced Admin web UI as XSA admin user.
2. Logon to the XS Advanced admin web UI with your XSA admin user.
3. Open the entry *Application Role Builder* and perform the following steps to add a new role collection.
   1. On the left, select the entry for *Role Collection* and select + to add and create a new role collection.
   2. Select the role collection, which you just created.
   3. Select the *Roles* tab and add an *Application Role*.
   4. Select *Fileprocessor Application* (use the application name determined by the command `xs env fileprocessor-web`).

5. Select the *Admin* template.
6. Select *Admin* role.
7. Save the role collection.
8. Return to the home page.

4. Create a new user.
   1. Open the User Management.
   2. Choose *New User*.
      Fill out the form and enter `<User Name>`, `<First Name>`, `<Last Name>`, `<Email>`, `<Password>` and choose *Create*.
   3. Select the *Role Collections* tab and add the newly created role collection.
   4. Choose *Save*.

3.2 Verifying the Setup

Context

After installation and setup of SAP File Processing, you should verify the installation.

Procedure

1. Open the following URL: `<SAP File Processing endpoint URL>/api/v4`.

   > **Note**
   >
   > You have to change the initial password of the user when you logon for the first time. Otherwise you will receive a "403 not authorized" response.
   >
   > To change the initial password use the endpoint URL which will redirect you to change the initial password.

2. Log on with the newly created user.

   > **Note**
   >
   > The user must have permission to access the service.

   The service response is in JSON format.

   You can use a browser plug-in for a convenient display of the JSON format.

4 Running the Application Smoke Test

SAP File Processing provides an API to developers. The application smoke test ensures that the basic API is working as expected.

Prerequisites

You completed installation, setup and configuration of SAP File Processing.

Context

SAP File Processing comes with a toolset (swagger and swagger UI) that allows to issue API calls (HTTP REST) against the SAP File Processing API interactively.

The test will mainly take place in the swagger UI.

Procedure

1. Access the Swagger UI.
2. Run the application checks.

Related Information

Accessing the Swagger UI [page 12]
Running Application Checks [page 12]

4.1 Accessing the Swagger UI

Prerequisites

We recommend to use a browser with a JSON viewer plug-in, as the service response is in JSON.

Procedure

1. Access the API root.

   Use the endpoint URL + `/api/v4` to access the API root. The endpoint URL is the URL of the SAP File Processing web application.
2. Start the swagger UI.

   SAP File Processing comes with a REST API test environment - the swagger UI. The swagger UI is used to issue API calls against SAP File Processing from the browser. Use the `swaggerUI` URL from api version `v4`.

   Example: `https://myserver:5000/swagger-ui.html`
3. Expand the service operations of the `onPremise` section.

   A service operation is an HTTP method on a URL path.

4.2 Running Application Checks

The application checks are minimal tests that check database access, job creation and successful file processing.

Perform the tests in the following order:

1. Check the Database Access [page 13]
2. Check the Job Creation [page 13]
3. Check the File Processing [page 15]

4.2.1 Check the Database Access

Context

Use the operation `GET v4/jobs` to determine, if the SAP HANA database is available.

Procedure

Click the service operation `GET v4/jobs` and then on the button *Try it out*.

Results

You get the response code 200 and the following response body:

Sample Code

```
{
    "links": {
      "first": "https://server:51064/api/v4/jobs?top=10&skip=0",
      "status": "https://server:51064/api/v4/jobs/status"
    },
    "totalJobCount": 0,
    "jobs": []
}
```

4.2.2 Check the Job Creation

Procedure

1. Use the operation `POST v4/jobs` to create a new job.

   Provide the following body:

   Sample Code

   ```
   {
      "id": "J001",
   ```

```
    "description": "My first file processing job"
}
```

The expected result: response code 201 and the following response body:

Sample Code

```
{
    "links": {
      "job": "https://server:51064/api/v4/jobs/J001"
    }
}
```

2. To verify the new job (jobId = J001) use the operation `GET v4/jobs/{jobid}`.

   The expected result:

   Response code 200 and a similar response body:

Sample Code

```
{
  "links": {
    "tasks": "https://server:51064/api/v4/jobs/J001/tasks",
    "status": "https://server:51064/api/v4/jobs/J001/status",
    "jobs": "https://server:51064/api/v4/jobs"
  },
  "id": "J001", "description": "My first file processing job",
  "isPaused": false,
  "active": true,
  "maxWorkers": 1,
  "changedAt": "2016-09-19T16:00:45.443Z",
  "changedBy": "BORSTENSON",
  "createdTs": "2016-09-19T16:00:45.000Z",
  "taskList": "J001",
  "rank": 0,
  "processTemplate": {
    "id": "defaultOnPremise",
    "description": "The standard process template for file processing with
extraction core",
    "steps": [
      {
        "id": "FileLoader",
        "description": "Loads the file for processing using HTTP",
        "nextStep": "BinaryToText"
      },
      {
        "id": "BinaryToText",
        "description": "Extracts the plaintext and text analysis from the
file"
      }
    ]
  }
}
```

Next Steps

The job was created and the file processing can be tested.

4.2.3 Check the File Processing

Prerequisites

At least one task had been created to test the file processing.

Context

> **Note**
>
> Note that the maximum file size for processing is 500 Megabyte (MB).

Procedure

1. Open a new browser window and use the following URL to display the current status of the job: `api/v4/jobs/J001/status`.
2. Choose *Start Auto Refresh*.
3. Switch to the Swagger UI browser window.
4. Use the operation `POST v4/jobs/{jobId}/tasks` to create a new task (`jobId = J001`).
5. Provide the following body:

   Sample Code

   ```
   [
     {
       "url": "https://someserver/files/CustomerContract_99826.pdf"
     }
   ]
   ```

   You can also use other resources on the internet if this URL is not available in your network. However if the URL cannot be accessed, the task reflects this in an error message.

   Expected result: response code 207 and the following response body (with a different task ID):

   Sample Code

   ```
   {
     "results": [
       {
         "id": 3042400378264,
         "status": 201,
         "links": {
   ```

```
        "task":
         "https://server:51064/api/v4/jobs/J001/tasks/3042400378264"
      }
    }
  ],
  "count": {
    "success": 1,
    "failed": 0
  }
}
```

The status of the task will change from *Running* into *Success*.

6. Verify the file processing.

   Use the operation `GET v4/jobs/{jobId}/tasks/{taskId}` to get the task details

   `jobId = J001,taskId = 3042400378264` (ID from the example above)

   Expected result: response code 200 and a similar response body:

   Sample Code

```
{
  "id": 3042400378264,
  "status": "BinaryToTextSuccess",
  "statusCategory": "success",
  "links": {
    "self": "https://server:51064/api/v4/jobs/J001/tasks/3042400378264",
    "plainText": "https://server:51064/api/v4/jobs/J001/tasks/3042400378264/
plaintext",
    "thumbnail": "https://server:51064/api/v4/jobs/J001/tasks/3042400378264/
thumbnail",
    "icon": "https://server:51064/api/v4/jobs/J001/tasks/3042400378264/
icon",
    "textAnalysis": "https://server:51064/api/v4/jobs/J001/tasks/
3042400378264/textanalysis",
    "SAPUniView": "https://server:51064/api/v4/jobs/J001/tasks/
3042400378264/suv",
    "tasks": "https://server:51064/api/v4/jobs/J001/tasks",
    "file": "http://fileloader-data.mo.sap.corp:8080/docs/File%20Search/
Technology/OMG/formal-08-04-08.pdf"
   },
   "error": {
     "code": "TEXT ANALYSIS: OK"
   },
   "mimeType": "application/pdf",
   "fileMD5": "f7ea8e682e0cfc031b23ab16ff14a315",
   "language": "en", "sizeFile": 1114572,
   "sizePlainText": 198334,
   "startTimeStamp": "2016-09-19T16:16:52.902Z",
   "attributes": {
     "language": "en",
     "contentType": "application/pdf",
     "extension": "pdf"
   },
   "subTasksCount": 0
}
```

7. Use the operation `GET v4/jobs/{jobId}/tasks/{taskId}/plaintext` to get the plaintext of the file:

   `jobId = J001,taskId = 3042400378264` (ID from the example above)

   Expected result: response code 200 and the plain text of the file.

5 HTTP Destinations

SAP File Processing uses GET requests to HTTP URLs to load the file content.

HTTP servers may not be available from the server network configuration. SAP File Processing supports the configuration of HTTP destinations to enable the use of a proxy server and basic authentication.

The destinations have to be maintained by the SAP File Processing administrator.

SAP File Processing will use the destinations for URLs that start with the destination URL.

Related Information

Creating HTTP Destinations [page 17]
Deleting HTTP Destinations [page 18]

5.1 Creating HTTP Destinations

Procedure

1. Open the SAP File Processing administration UI in the browser.

   Use the root URL of the SAP File Processing service.
2. Select the *Destinations* tile and click *+ Create*.
3. Fill out the form and press *OK* to save the HTTP destinations.

5.2 Deleting HTTP Destinations

Procedure

1. Open the SAP File Processing administration UI in the browser.

   Use the root URL of the SAP File Processing service.
2. Select the *Destinations* tile and click *Delete*.

6 Using the SAP File Processing API UI (Swagger UI)

Context

SAP File Processing comes with a test environment for the API. As an administrator you can invoke the SAP File Processing API UI.

Refer to the swagger documentation as the UI is based on the Swagger UI. See `http://swagger.io` for details.

Procedure

Open the SAP File Processing API UI.

Click the documentation link in the SAP File Processing Administration UI or open the following HTML page under the system URL: `/swagger-ui.html`.

7 Scaling Master and Worker Applications

The master application and the web application can be scaled depending on your scenario.

Scale the master application to handle API load. One master application instance is deployed by default.

Scale the worker application to scale the processing speed of the files. Three workers are deployed by default. There can be a maximum of 10 worker apps.

The system administrator can use the XSA command line client to scale the applications using the scale command.

Sample Code

Start two master application instances

```
xs scale fileprocessor-master -i 2
```

8 Security

SAP File Processing uses the standard XS advanced security setup.

It uses UAA for authentication and HDI container for data separation.

An XSA administrator creates role collections, users and role assignments in the XSA Administration.

For more information, refer to the XS advanced security documentation.

Related Information

Roles [page 21]

8.1 Roles

SAP File Processing uses the following roles:

Table 2:

Role Name	Description
API	The *API* role should be applied to the technical user that are used by the client application. This user role provides the authorization to invoke all exposed HTTP service operations of the API. Path to the API: `/api/v4` HTTP methods: GET, PUT, POST, DELETE
Auditor	The *Auditor* role should be applied to a user with read only API access. This user role provides read only access to the exposed HTTP service operations of the API. Path to the API: `/api/v4` HTTP methods: GET
Administrator	The *Administrator* role should be applied to a user who is using the administration UI and the API UI. The role includes all authorizations of the *API* role.

9 SAP File Processing API

The SAP File Processing API is an HTTP REST API.

The following documentation lists all supported service operations and the used models (data structures).

In addition to the documented HTTP status codes, the API client should handle the following standard HTTP codes:

- 304: Not Modified
- 401: Unauthorized – Authentication failed
- 403: Forbidden – Authorization failed
- 405: Method Not Allowed
- 500: Internal Server Error
- 502: Bad Gateway
- 503: Service Unavailable
- 504: Gateway Timeout1

The base path of the API is `/api`.

9.1 Supported Service Operations

The API supports the following service operations:

- `DELETE /v4/destinations/{destinationId}`
  Deletes a single HTTP destination.
- `GET /v4/destinations/{destinationId}`
  Returns a single HTTP destination.
- `GET /v4/destinations/{destinationId}/test`
  A GET request is issued against the destination url to test the accessibility of the server, the authentication data and the proxy data.
- `GET /v4/destinations`
  Get all defined HTTP destinations to access HTTP resources using basic authentication with proxy server support.
- `POST /v4/destinations`
  Creates a new HTTP destination.
- `GET /v4/destinations/test`
  Tests if an URL can be accessed by the system. The system may use a destination if one is defined for the URL.
- `GET /v4`
  The root service returns general information of the SAP File Processing service. General meta information of the SAP File Processing REST service. The service provides version information of the component and supported API version. The service is used by clients to determine general version dependencies.

- `GET /v4/jobs`
  Returns a list of jobs.
- `PUT /v4/jobs/{jobId}/activate`
  Activates a job.
- `DELETE /v4/jobs/{jobId}`
  Delete a specific job.
- `GET /v4/jobs/{jobId}`
  Returns the job specified by the ID.
- `PUT /v4/jobs/{jobId}/pause`
  Pauses a specific job.
- `PUT /v4/jobs/{jobId}`
  Updates the job specified by the ID.
- `GET /v4/jobs/{jobId}/status`
  Returns current status information of the job.
- `DELETE /v4/jobs/{jobId}/tasks`
  Deletes multiple tasks.
- `GET /v4/jobs/{jobId}/tasks`
  Returns a list of tasks of the specified job.
- `PUT /v4/jobs/{jobId}/tasks/harvest/package`
  Returns a package of tasks and marks them as harvested.
- `PUT /v4/jobs/{jobId}/tasks/harvest/reset`
  Resets harvested flag on all tasks.
- `DELETE /v4/jobs/{jobId}/tasks/{key}`
  Deletes a single task.
- `GET /v4/jobs/{jobId}/tasks/{key}`
  Returns the details of a single task.
- `GET /v4/jobs/{jobId}/tasks/{key}/plaintext`
  Returns the plaintext as text of a single task, if the text conversion was successful.
- `POST /v4/jobs/{jobId}/tasks`
  Creates many new tasks. The response describes if the tasks creation was successful or failed.
- `POST /v4/jobs`
  Creates a new job.
- `GET /v4/jobs/status`
  Provides summary information of all jobs.
- `GET /v4/mimeTypes`
  Returns a list of supported mime types.
- `GET /v4/swagger`
  Returns the API definition.
- `DELETE /v4/workers`
  Deletes worker events.
- `GET /v4/workers`
  Returns a list of workers.
- `GET /v4/workers/{workerId}/events`
  Get worker events.
- `GET /v4/workers/{workerId}`
  Get the worker details.

Important Disclaimers and Legal Information

Coding Samples

Any software coding and/or code lines / strings ("Code") included in this documentation are only examples and are not intended to be used in a productive system environment. The Code is only intended to better explain and visualize the syntax and phrasing rules of certain coding. SAP does not warrant the correctness and completeness of the Code given herein, and SAP shall not be liable for errors or damages caused by the usage of the Code, unless damages were caused by SAP intentionally or by SAP's gross negligence.

Accessibility

The information contained in the SAP documentation represents SAP's current view of accessibility criteria as of the date of publication; it is in no way intended to be a binding guideline on how to ensure accessibility of software products. SAP in particular disclaims any liability in relation to this document. This disclaimer, however, does not apply in cases of willful misconduct or gross negligence of SAP. Furthermore, this document does not result in any direct or indirect contractual obligations of SAP.

Gender-Neutral Language

As far as possible, SAP documentation is gender neutral. Depending on the context, the reader is addressed directly with "you", or a gender-neutral noun (such as "sales person" or "working days") is used. If when referring to members of both sexes, however, the third-person singular cannot be avoided or a gender-neutral noun does not exist, SAP reserves the right to use the masculine form of the noun and pronoun. This is to ensure that the documentation remains comprehensible.

Internet Hyperlinks

The SAP documentation may contain hyperlinks to the Internet. These hyperlinks are intended to serve as a hint about where to find related information. SAP does not warrant the availability and correctness of this related information or the ability of this information to serve a particular purpose. SAP shall not be liable for any damages caused by the use of related information unless damages have been caused by SAP's gross negligence or willful misconduct. All links are categorized for transparency (see: http://help.sap.com/disclaimer).

go.sap.com/registration/contact.html