

Using Snaplogic to Data Extraction

This guide provides a step-by-step approach to using SnapLogic for extracting data from various formats such as PDFs, CSVs, JSON, and HTML. Whether you are new to SnapLogic or looking to refine your data extraction processes, this guide will help you create and manage effective pipelines.

Data Formats in SnapLogic

SnapLogic is capable of reading data from a wide range of sources and can convert this data into two primary formats:

1. Document Format:

- A document in SnapLogic is a JSON-like format used for internal data processing. This format allows users to interact with data attributes in a manner similar to JSON. It also supports SnapLogic's internal functions, such as text processing, enabling users to apply expressions for data manipulation and transformation. These transformations can be performed before the data is sent to downstream connectors, ensuring that the data is appropriately structured and formatted for subsequent processing.

2. Binary Format:

- The binary format in SnapLogic is used to handle raw data streams, such as images, PDFs, or any other file types that do not conform to structured data formats. This format is particularly useful when dealing with non-text data or when the data needs to be passed through the pipeline without alteration. SnapLogic can convert binary data to other formats when necessary, allowing for processing, storage, or transformation before being transmitted to downstream components. This ensures flexibility and efficiency in handling various types of data within the pipeline.

Extracting Data from Files in SnapLogic

SnapLogic offers robust capabilities for extracting data from various file types, making it a versatile tool in data integration workflows. To facilitate this process, SnapLogic provides the **SnapLogic File System (SLFS)**, a built-in storage solution where users can upload and manage files directly within the SnapLogic environment. However, there are important considerations and best practices to keep in mind when utilizing SLFS for data extraction.

Using SnapLogic File System (SLFS)

The SnapLogic File System (SLFS) allows users to store and access files for processing within their pipelines. To extract data from a file using SnapLogic, users first need to upload the

file to SLFS. Once uploaded, the file can be accessed and processed by various Snaps designed to read and manipulate file content.

However, SLFS has a file size limitation, allowing users to upload files up to a maximum of **100MB** per file. This limitation can present challenges when dealing with larger datasets or files. If the file exceeds this size limit, users would need to split the content into multiple smaller files before uploading them to SLFS. While this approach may work for smaller or segmented data, it is generally not recommended for handling large-scale data processing tasks.

Best Practices for Handling Large Files

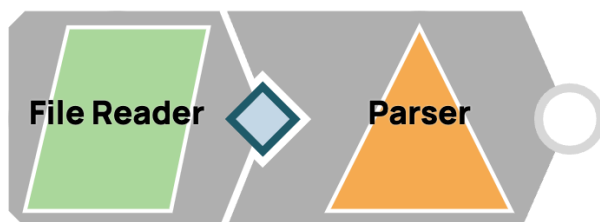
For processing files larger than 100MB, it is advisable to leverage external storage systems rather than relying on SLFS. External file systems provide greater flexibility and scalability, allowing you to handle significantly larger files without the need for manual segmentation. SnapLogic seamlessly integrates with various external storage solutions, enabling efficient data extraction and processing.

- **SFTP Servers:** Users can utilize Secure File Transfer Protocol (SFTP) servers for storing and accessing large files. SnapLogic supports SFTP integration, allowing you to securely read and process files directly from the server within your pipelines.
- **Object Storage Services (e.g., Amazon S3):** Cloud-based object storage services like Amazon S3 offer virtually unlimited storage capacity, making them ideal for managing large datasets. SnapLogic's native support for S3 allows users to read, write, and manage files stored in S3 buckets, ensuring smooth data processing without the constraints of SLFS.

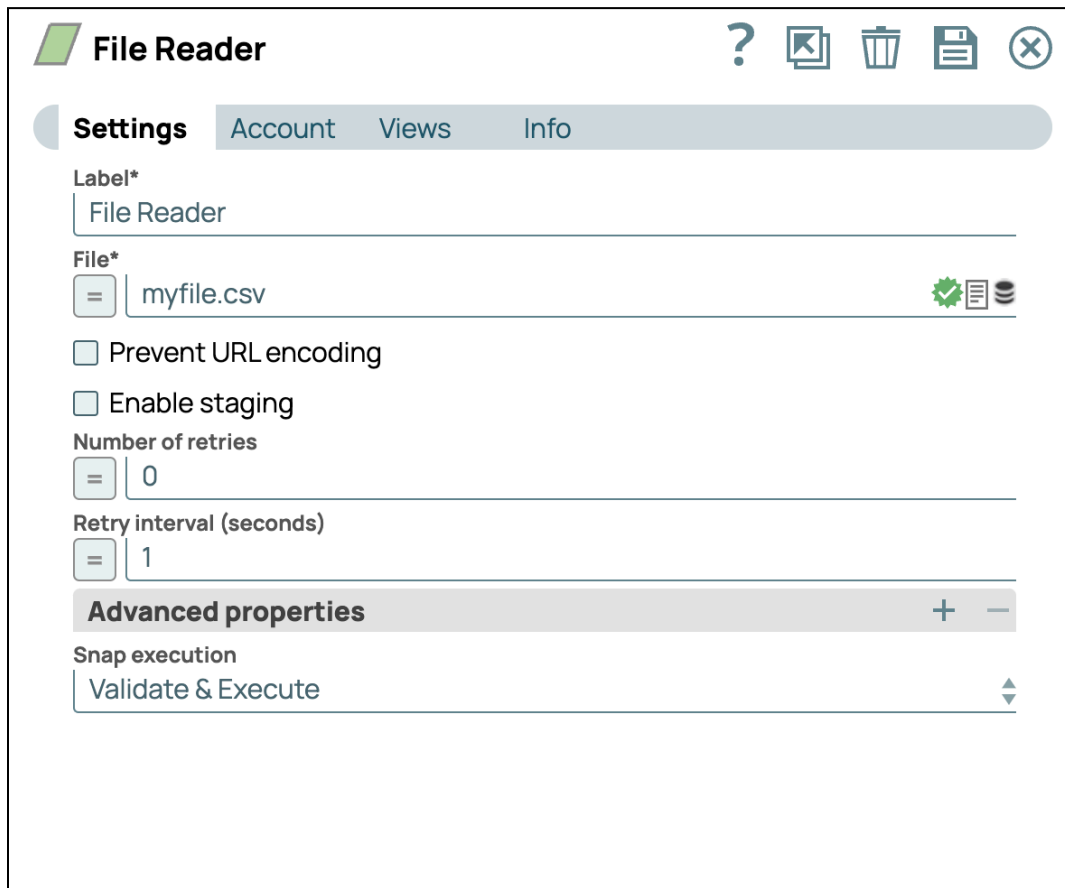
By integrating SnapLogic with these external storage solutions, users can efficiently manage large files, streamline data extraction workflows, and maintain optimal performance across their pipelines. This approach not only overcomes the limitations of SLFS but also aligns with best practices for scalable and reliable data processing.

Reading File Data and process in SnapLogic

In this section, we'll delve into the process of reading data from various file formats using SnapLogic. This guide outlines the key steps, configuration options, and best practices to ensure efficient and effective data extraction. To read data from a file using SnapLogic, you can follow these two essential steps:



1. Configuring the File Reader Snap



The screenshot shows the configuration interface for a 'File Reader' Snap. At the top, there is a title bar with a green icon and the text 'File Reader', followed by icons for help, zoom, delete, save, and close. Below the title bar is a tabbed menu with 'Settings' selected, and other tabs for 'Account', 'Views', and 'Info'. The 'Settings' section includes a 'Label*' field with the value 'File Reader', a 'File*' field with the value 'myfile.csv' and a file icon, and two checkboxes: 'Prevent URL encoding' and 'Enable staging', both of which are unchecked. Below these are two numeric input fields: 'Number of retries' with the value '0' and 'Retry interval (seconds)' with the value '1'. At the bottom, there is an 'Advanced properties' section with a plus and minus icon, and a 'Snap execution' dropdown menu with the value 'Validate & Execute'.

Configuring the File Reader Snap is crucial for setting up a SnapLogic pipeline that can accurately and efficiently access data from various file systems and storage solutions. This Snap serves as the gateway for bringing data into your pipeline, and its proper configuration is key to seamless data extraction.

Key Considerations:

- **Identifying the File Location:**
 - SnapLogic supports a wide range of file storage protocols, enabling access to files stored both locally and in remote environments
 - **SLFS (SnapLogic File System):** Ideal for smaller files stored temporarily within SnapLogic.
 - **Amazon S3:** A scalable cloud storage service best managed with the **S3 File Reader Snap** for optimized performance and ease of use.
 - **SFTP Servers:** For secure file transfers over a network using SFTP.
 - **HTTP/HTTPS:** Allows direct access to files hosted on web servers.
 - **Azure Blob Storage:** Optimized for handling large amounts of unstructured data in the cloud.

- **Selecting the Appropriate Protocol:**

Choosing the correct protocol ensures efficient file access based on the storage location. Each protocol requires specific configurations to enable secure and accurate data retrieval. Below are the guidelines for configuring various storage protocols.

- **SLFS:** Specify the file path relative to the SLFS root.
- **Amazon S3:** It is recommended to use the **S3 File Reader Snap** for S3-specific features. If using the File Reader Snap, ensure the correct bucket name, file path, and AWS credentials are provided.
- **SFTP:** Input the server address, file path, and authentication details (username, password, or SSH key).
- **HTTP/HTTPS:** Enter the full URL, including any necessary authentication tokens.
- **Azure Blob Storage:** Provide the container name, blob path, and relevant credentials.

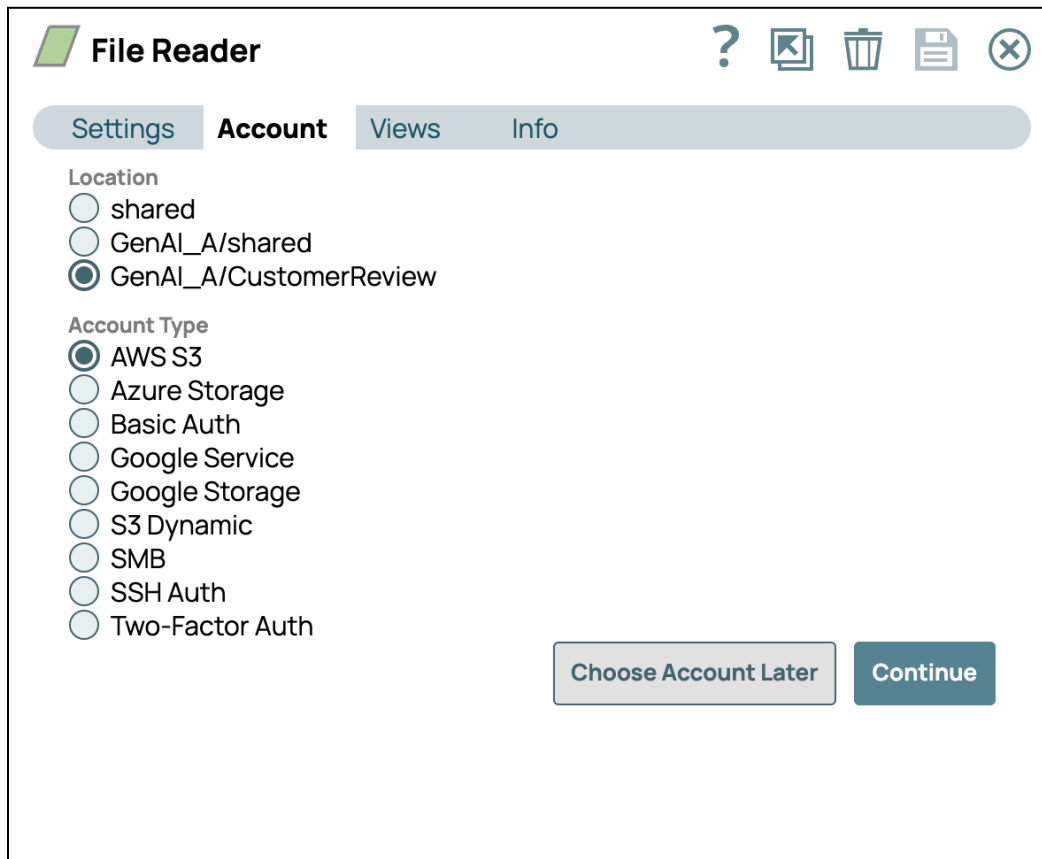
- **File Path Configuration:**

Properly formatting the file path is crucial for successful data access. Each protocol requires specific path structures

- **SLFS:** Use a relative path (e.g., `/myfiles/data.csv`).
- **S3:** Format the path for S3 buckets (e.g., `s3://mybucket/data/myfile.csv`), and consider using the **S3 File Reader Snap** for simplicity.
- **SFTP:** Provide the full path relative to the SFTP root (e.g., `/home/user/data/myfile.csv`).
- **HTTP/HTTPS:** Use the full URL (e.g., `https://www.example.com/files/myfile.csv`).
- **Azure Blob Storage:** Specify the blob path within the container.

- **Authentication and Credentials:**

Securely accessing external storage systems requires robust authentication. Depending on the storage type, different methods are used to ensure authorized access to sensitive data. Proper configuration of these methods is essential for maintaining data integrity and security.



- **S3:** Use AWS Access Key ID and Secret Access Key, or IAM roles for AWS environments. The **S3 File Reader Snap** simplifies this process.
- **SFTP:** Configure with username/password or SSH keys, ensuring the correct key file is specified.
- **HTTP/HTTPS:** Input basic authentication credentials or bearer tokens as needed.
- **Azure Blob Storage:** Use SAS tokens or AAD credentials depending on your security requirements.

2. Selecting the Appropriate Parser Snap

Once the File Reader Snap is configured, the next step is to select the appropriate Parser Snap for processing the file data:

- **CSV Files:** Use the **CSV Parser Snap** to read and parse comma-separated values (CSV) files. This Snap handles various delimiters and header rows, allowing direct mapping of CSV data to downstream processes.
- **JSON Files:** The **JSON Parser Snap** is ideal for parsing both flat and nested JSON structures, converting them into SnapLogic's Document format for further processing.
- **XML Files:** The **XML Parser Snap** effectively handles XML data, supporting the parsing of complex XML structures, including attributes and nested elements.

- **HTML Files:** Use the **HTML Parser Snap** to extract data from HTML documents, with support for XPath or CSS selectors to accurately pinpoint specific elements within the HTML.
- **PDF Files:** The **PDF Parser Snap** is designed to convert PDFs into a structured format like JSON, enabling further manipulation within SnapLogic.

Example of reading a file in Snaplogic

Reading CSV File



- 1) **Add the File Reader Snap:** Drag and drop the “File Reader” Snap onto the designer workspace.
- 2) **Configure the File Reader Snap:** Click on the “File Reader” Snap to access its settings panel.

The screenshot shows the settings panel for the 'File Reader' snap. The panel has a title bar with a green icon and the text 'File Reader', along with icons for help, zoom, delete, save, and close. Below the title bar are tabs for 'Settings', 'Account', 'Views', and 'Info'. The 'Settings' tab is active. The settings include:

- Label*:** A text input field containing 'File Reader'.
- File*:** A text input field containing 'review.csv' with a file selection icon to its right.
- Prevent URL encoding
- Enable staging
- Number of retries:** A text input field containing '0'.
- Retry interval (seconds):** A text input field containing '1'.
- Advanced properties:** A section header with expand/collapse icons.
- Snap execution:** A dropdown menu currently set to 'Validate & Execute'.

- 3) **Select or Upload a File:** In the file settings, click on the folder icon to either upload a new file or select an existing one from the directory.

Select File

Project: GenAI_A/CustomerReview ▼ ↻ Upload File

filter

File Path	File size	Last modified
/snaplogic/GenAI_A/CustomerReview/European Restaurant Reviews.csv	674KB	08/13/2024, 12:46 PM
/snaplogic/GenAI_A/CustomerReview/review.csv	134KB	08/13/2024, 01:13 PM

Showing 1 - 2 of 2 entries ◀◀ 1 of 1 ▶▶

- 4) **Save Configuration:** After selecting or uploading the desired file, click "Save" and then close the settings panel.
- 5) **Add the CSV Parser Snap:** Drag and drop the "CSV Parser" Snap onto the workspace, positioning it after the File Reader. The default settings for the CSV Parser are typically sufficient for most use cases.
- 6) **Validate or Execute the Pipeline:** Validate or execute the pipeline to process and read the contents of the CSV file.

Reading HTML Content with File Reader



- 1) **Add the File Reader Snap:** Drag and drop the “File Reader” Snap onto the designer workspace.
- 2) **Configure the File Reader Snap:** Click on the “File Reader” Snap to access its settings panel. Then put the url that you want to connect to the settings

File Reader1 ? [Zoom] [Delete] [Save] [Close]

Settings Account Views Info

Label*
File Reader1

File*
=<url> [Checkmark] [List] [Globe]

Prevent URL encoding

Enable staging

Number of retries
= 0

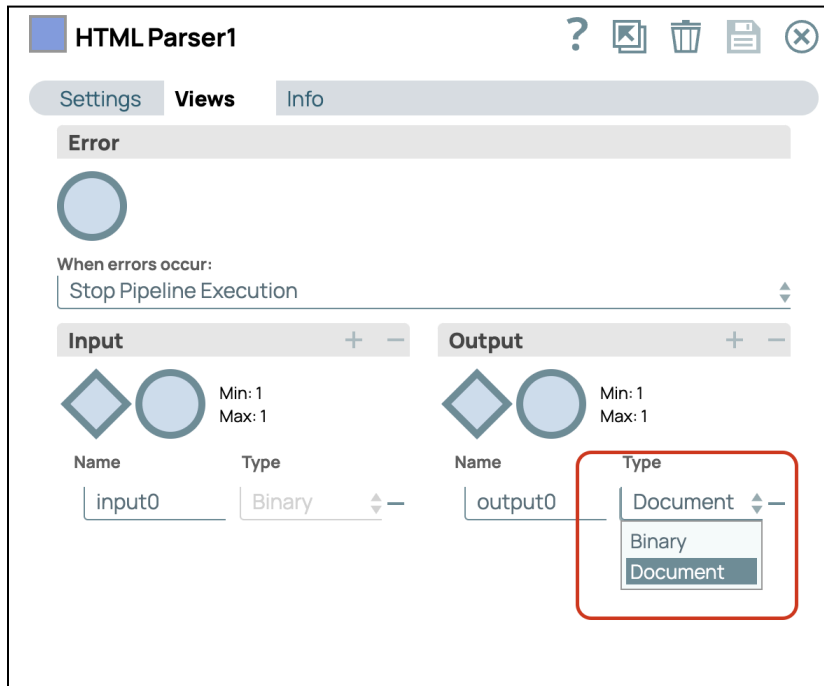
Retry interval (seconds)
= 1

Advanced properties + -

Snap execution
Validate & Execute

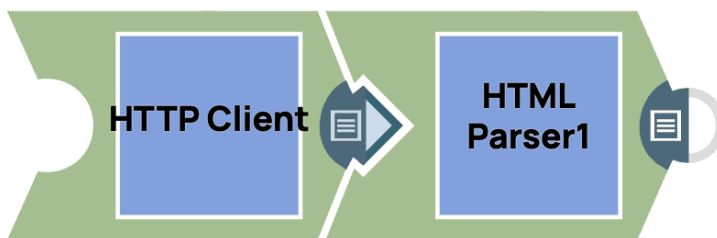
- 3) **Save Configuration:** Click "Save" and close the settings.
- 4) **Add the HTML Parser Snap:** Drag the "HTML Parser" snap to the Designer and connect it to the File Reader snap and leave the configuration as a default

- 5) **Configure the HTML Parser Snap:** Open the HTML Parser settings and go to the "Views" tab. Choose to output either as a document or binary.



- 6) **Validate or Execute the Pipeline:** Validate or execute the pipeline to retrieve the text content from the given URL.

Reading HTML Content with HTTP Client



- 1) **Add the HTTP Client Snap:** Drag and drop the "HTTP Client" Snap onto the designer workspace.
- 2) **Configure the HTTP Client Snap:** Click on the "HTTP Client" Snap to access its settings panel. Then put the url that you want to connect to the settings. The HTTP Client snap allows user to do complex html calling with various of configuration. Users can select "Request Methods" or provide the "Pagination" mechanism to get continuous content

HTTP Client

? [Icons]

Settings Account Views Info

Label*
HTTP Client

Request method*
GET

URI*
<url>

Query parameters + -

HTTP headers + -

Enable debug

Use form encoding for spaces

Prevent URL encoding

HTTP entity

Entity type
none

▼ Pagination

Has next
[=] [▼]

Total pages to fetch
[=] [_____]

Override URI

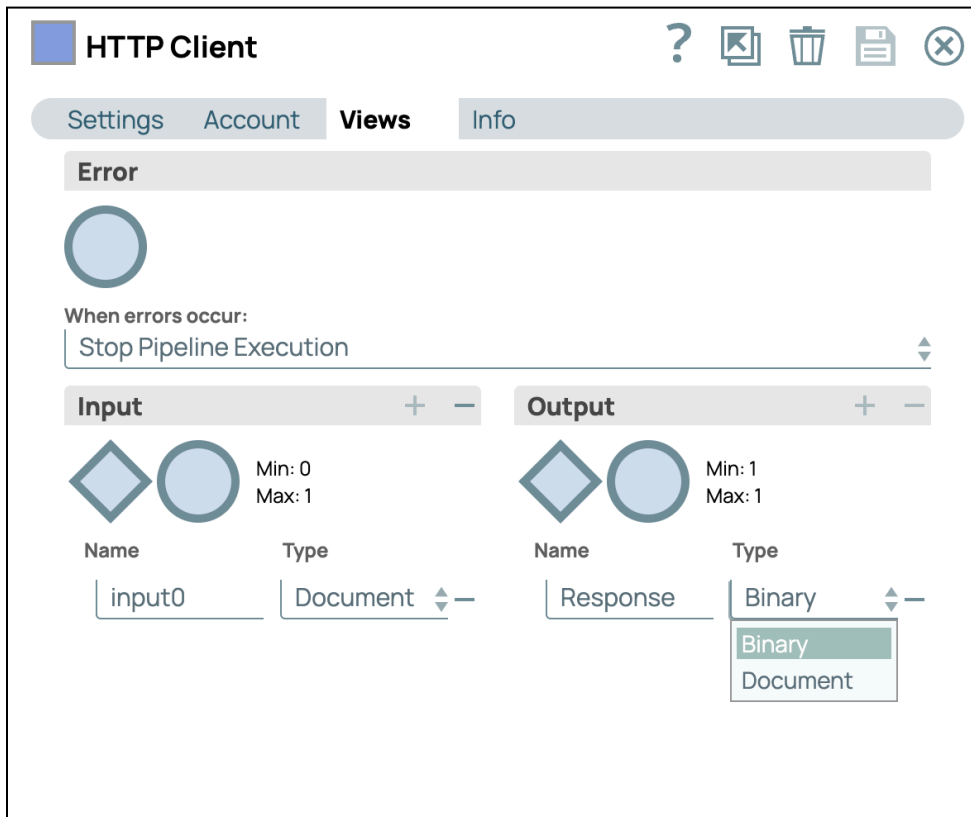
Reuse request parameters

Override headers + -

Name	Value
[=] [_____]	[=] [_____]

Pagination interval (seconds)
[_____]

- 3) **Configure the HTTP Client Snap Views:** Click the “Views” tab and select output to “Binary”. Then save and close the settings.



- 4) **Add the HTML Parser Snap:** Drag the "HTML Parser" snap to the Designer and connect it to the File Reader snap and leave the configuration as a default
- 5) **Validate or Execute the Pipeline:** Validate or execute the pipeline to retrieve the text content from the given URL.
- 6) **View output:** The output will show the content in text format



Reading PDF File

To effectively read and extract information from a PDF file, users must first understand the nature of the content within the file. PDF files can generally be classified into two categories: those containing text-based content and those containing image-based content.

For PDFs with text-based content, users can utilize a standard PDF parser, which is designed to extract and output the textual information in a readable format. These parsers are widely available and are efficient in converting the embedded text in a PDF document into a text format that can be further processed or analyzed.

However, if the PDF contains image-based content, such as scanned documents or images of text, a different approach is required. In this case, Optical Character Recognition (OCR) technology must be employed. OCR services are specialized tools that analyze the images within the PDF and convert the visual representations of text into machine-readable text. This process is crucial for making the content accessible and editable, especially when dealing with scanned documents or other image-heavy files.

By understanding these distinctions and choosing the appropriate tool for the type of content within a PDF, users can effectively extract the necessary information, ensuring accuracy and efficiency in their workflow.

Reading PDF File with PDF Parser snap



- 1) **Add the File Reader Snap:** Drag and drop the “File Reader” Snap onto the designer workspace.

- 2) **Configure the File Reader Snap:** Click on the “File Reader” Snap to access its settings panel. Then, select or upload a pdf file that you want to read

File Reader1

? [Copy] [Trash] [Save] [Close]

Settings Account Views Info

Label*
File Reader1

File*
[=] file.pdf [Check] [List] [Globe]

Prevent URL encoding

Enable staging

Number of retries
[=] 0

Retry interval (seconds)
[=] 1

Advanced properties + -

Snap execution
Validate & Execute [Dropdown]

- 3) **Save Configuration:** Click "Save" and close the settings.
- 4) **Add the PDF Parser Snap:** The PDF Parser snap can parse the pdf file into a text.

- 5) **Configure the PDF Parser Snap:** Users can select a proper “Parser type” to parse pdf file. In this case, we select “Text extractor” so we can get all text from pdf file.

PDF Parser ? [Zoom] [Trash] [Print] [Close]

Settings Account Views Info

Label*
PDF Parser

Pages
=

Parser type*
Text extractor

Snap
Val
Text extractor
Images extractor from pages
Pages to images converter
Table parser

- 6) **Validate or Execute the Pipeline:** Validate or execute the pipeline to retrieve the text content from the given PDF file. The response contains pdf text that are extracted from the pdf file.

PDF Parser output0

Preview Type Indent Level Expand Level
JSON 2 1+

```
▼ [
  ▼ {
    "text": "S20 Diabetes Care Volume 47, Supplement 1, January 2024 2. Diagnosis and Classification of Ameri
  }
]
```

Reading PDF File with Unstructured



Prerequisite: Users need to have unstructured API key to access the unstructured service or have the unstructured instance deploy locally to use the unstructured API

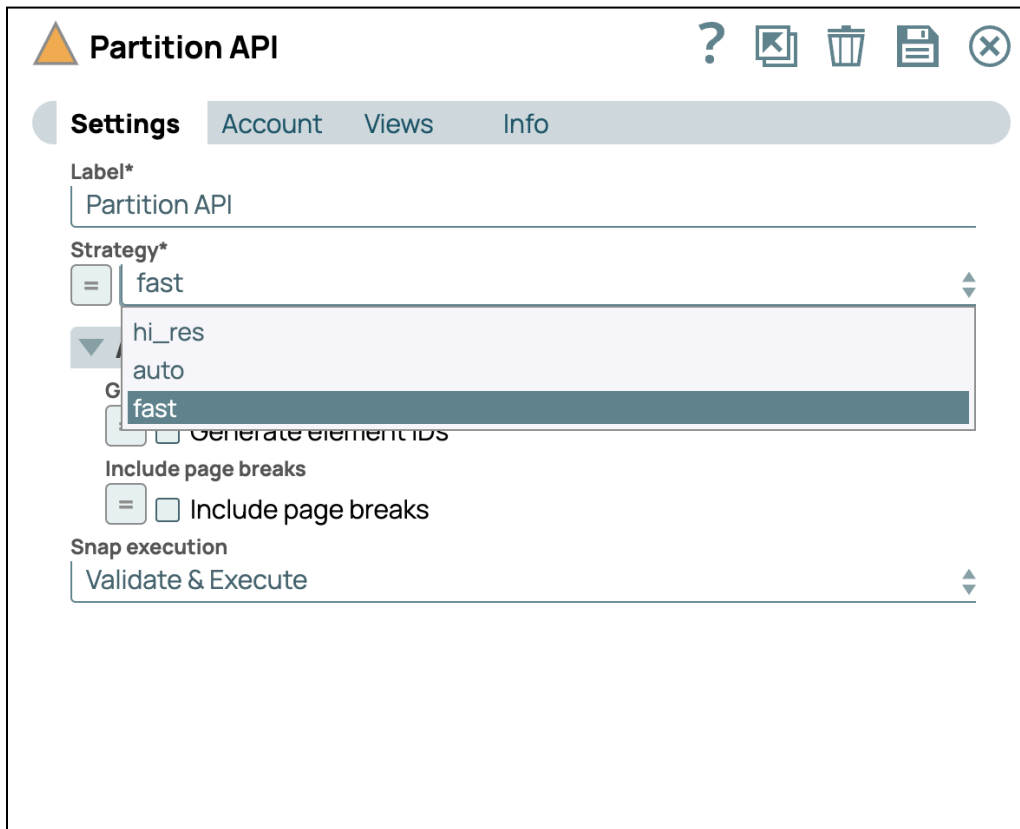
- 1) **Add the File Reader Snap:** Drag and drop the “File Reader” Snap onto the designer workspace.
- 2) **Configure the File Reader Snap:** Click on the “File Reader” Snap to access its settings panel. Then, select or upload a pdf file that you want to read

The screenshot shows the settings panel for a 'File Reader1' snap. At the top, there are icons for help, zoom, delete, save, and close. Below the title bar, there are tabs for 'Settings', 'Account', 'Views', and 'Info'. The 'Settings' tab is active and contains the following fields:

- Label*:** A text input field containing 'File Reader1'.
- File*:** A file selection field containing 'file.pdf' with a green checkmark icon and a document icon.
- Prevent URL encoding
- Enable staging
- Number of retries:** A numeric input field containing '0'.
- Retry interval (seconds):** A numeric input field containing '1'.
- Advanced properties:** A section header with expand/collapse icons.
- Snap execution:** A dropdown menu currently set to 'Validate & Execute'.

- 3) **Save Configuration:** Click "Save" and close the settings.

- 4) **Add the Partition API Snap:** The partition API snap utilize the unstructured partition API to parse a pdf file.



The screenshot shows the configuration interface for the Partition API. At the top, there is a title bar with an orange triangle icon and the text "Partition API". To the right of the title bar are several icons: a question mark, a window icon, a trash can, a document icon, and a close button (X). Below the title bar is a navigation bar with tabs for "Settings", "Account", "Views", and "Info". The "Settings" tab is selected. The main content area contains the following configuration options:

- Label*:** A text input field containing "Partition API".
- Strategy*:** A dropdown menu with a list of options: "fast", "hi_res", "auto", and "fast". The "fast" option is currently selected.
- Generate element IDs:** A checkbox that is currently unchecked.
- Include page breaks:** A checkbox that is currently unchecked.
- Snap execution:** A dropdown menu with the option "Validate & Execute" selected.

- 5) **Configure the Partition API Snap:** Users can select strategy to parse a pdf file. In this case, we use "fast" strategy to parse only text and not include image and table. If you select hi_res, the image and table will be parsed as a base64.
- 6) **Validate or Execute the Pipeline:** Validate or execute the pipeline to retrieve the text content from the given PDF file. The response also contains type of the content that are parsed from the PDF. So users can use "type" attribute to process further for example,

users can process title and content separate from the footer.

```
Partition API output0

Preview Type  Indent Level  Expand Level
JSON         2             1+

▼ [
  ▼ {
    "type": "Header",
    "element_id": "1c7694059e1e0b694ac7d5df61ff9990",
    "text": "S20",
    ▶ "metadata": { "languages": [...], "page_number": 1, "filename": "dc24s002.pdf", "filetype": "application/pdf",
    ▶ "original": { "content-type": "application/pdf", "date": "Thu, 15 Aug 2024 07:07:47 GMT", "server": "istio-envoy", "c..."
  },
  ▶ {"type": "Header", "element_id": "b01b108cfe78227e8a73b00a7e4b4a3d", "text": "Diabetes Care Volume... },
  ▼ {
    "type": "Title",
    "element_id": "02c74fc427706f3050bd3de1f5accebb",
    "text": "S e A T e B A D",
    ▶ "metadata": { "languages": [...], "page_number": 1, "parent_id": "b01b108cfe78227e8a73b00a7e4b4a3d", "filename": "d...",
    ▶ "original": { "content-type": "application/pdf", "date": "Thu, 15 Aug 2024 07:07:47 GMT", "server": "istio-envoy", "c..."
  },
  ▶ {"type": "Title", "element_id": "e334ea37fbaf5626bdb26a6f76a48c1a", "text": "I", "metadata": { "...", "original... },
  ▶ {"type": "Title", "element_id": "991325a1770126422a7aacfc2ea22a48", "text": "F O", "metadata": { "...", "orig... },
  ▼ {
    "type": "UncategorizedText",
    "element_id": "7f9755201b80c954762c25893a2c5e5f",
    "text": "N o l e T e A c e l e F a l e S e A e L e C",
    ▶ "metadata": { "languages": [...], "page_number": 1, "parent_id": "991325a1770126422a7aacfc2ea22a48", "filename": "dc...",
    ▶ "original": { "content-type": "application/pdf", "date": "Thu, 15 Aug 2024 07:07:47 GMT", "server": "istio-envoy", "c..."
  },
  ▶ {"type": "Title", "element_id": "f9cdc4fe0b4ac56c408a263237fe2ef3", "text": "D e N e A", "metadata": { "...", "orig... },
  ▶ {"type": "Title", "element_id": "6a03ad475ac657e0e5710fd9f0e0e0", "text": "S e L e C A D", "text" }
```

Reading PDF File with Adobe Service

Users can leverage the Adobe API for PDF parsing tasks. SnapLogic offers the **Adobe Extract Snap** for extracting text from PDF documents, as well as the **Adobe OCR Snap** for processing image-based PDF content. When dealing with PDFs that contain image-based content, the Adobe OCR Snap is the appropriate tool, enabling accurate extraction of text from images within the document.

Reading PDF File with Adobe OCR



Prerequisite: Users need to have Adobe key to access the Adobe OCR service before using the Adobe OCR snap

- 1) **Add the File Reader Snap:** Drag and drop the “File Reader” Snap onto the designer workspace.

- 2) **Configure the File Reader Snap:** Click on the “File Reader” Snap to access its settings panel. Then, select or upload a pdf file that you want to read

File Reader1

? [Zoom] [Delete] [Save] [Close]

Settings Account Views Info

Label*
File Reader1

File*
file.pdf [Checkmark] [Document] [Globe]

Prevent URL encoding

Enable staging

Number of retries
0

Retry interval (seconds)
1

Advanced properties + -

Snap execution
Validate & Execute

- 3) **Save Configuration:** Click "Save" and close the settings.
- 4) **Add the Adobe OCR Snap:** The Adobe OCR Snap utilizes the Adobe API to efficiently parse PDF files with image-based content. Users can rely on the default settings provided within the Adobe OCR Snap to accurately extract text from images embedded in the PDF.
- 5) **Validate or Execute the Pipeline:** Validate or execute the pipeline to retrieve the text content from the given PDF file. The response contains a binary content that contains the extracted information.