SmartRead

SmartRead API v3 概要

2024/08/16

株式会社 Cogent Labs

COGENT LABS

SHAPING THE FUTURE
WITH ARTIFICIAL INTELLIGENCE

本資料について

本資料は、2024年7月22日時点の「SmartRead API v3」の概要紹介資料です。
APIドキュメントの補足資料としてご参照ください。

以下の留意点をあらかじめご確認ください。

- 「APIドキュメント」と内容について差分がある場合、「APIドキュメント」の内容を正として使用してください。
- SmartRead On-Premises 固有の管理系APIについては紹介していません。
  マニュアル「SmartRead On-Premises API ご利用ガイド」を参照ください。
- **「SmartRead非定型文書強化バージョン」用API（v4）は紹介していません。**

目次

01　SmartRead API とは

02　事前準備

03　SmartRead API v2 使用時の留意点

04　定型文書の読取処理の実行例（Shell+Curl）

05　読取処理の自動化

参考　用語の解説

01 SmartRead API とは

SmartRead API の目的

SmartRead Webページより

SDKは提供していません。

SmartRead API のバージョン

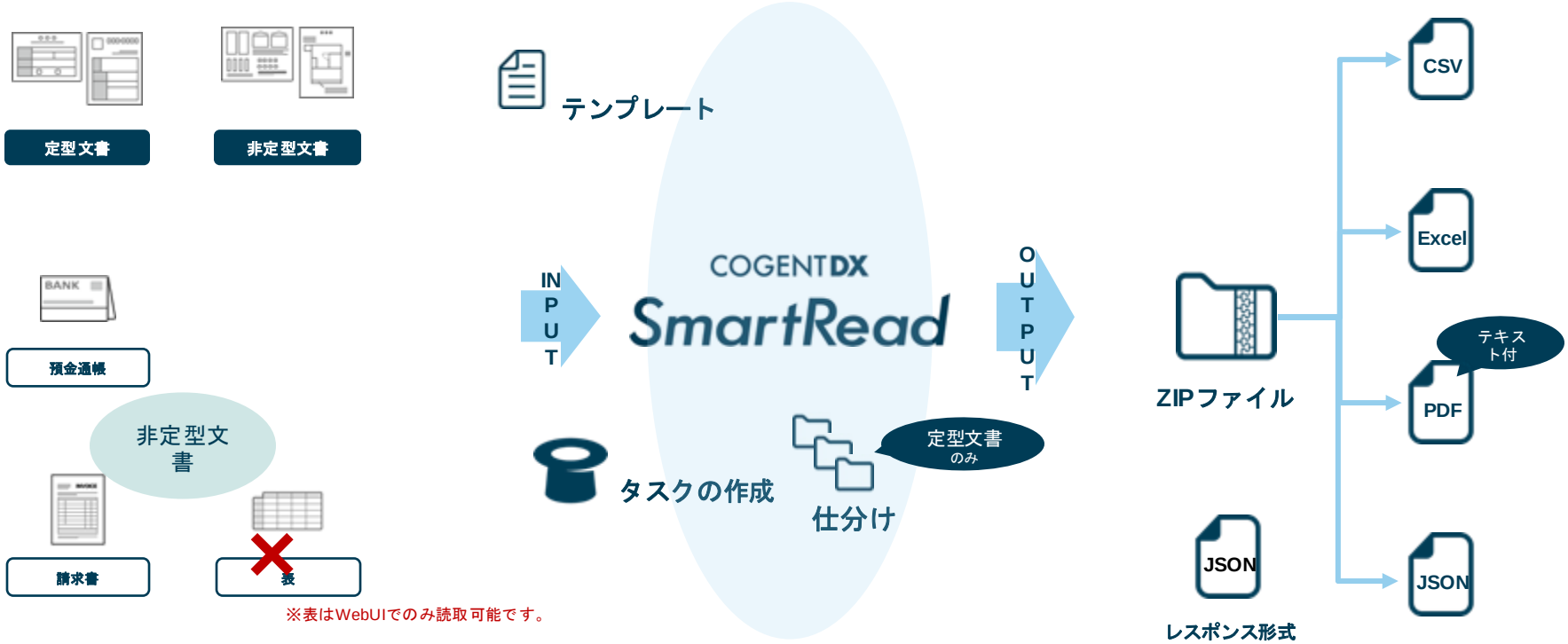
- APIの種類
  - REST API
- バージョン
  - Version 4
    - 「SmartRead非定型文書強化バージョン」で使用可能
  - Version 3
    - 「SmartRead」で使用可能
    - サポート終了予定無し
  - Version 2
    - 「Tegaki」、「SmartRead」 で使用可能
    - サポート終了予定無し

「REST API」が呼び出せる開発言語であれば、
プログラム開発は可能です。

製品と使用できるAPIバージョン

製品	v2	v3	v4
SmartRead非定型文書強化バージョン			○
SmartRead	○	○	
Tegaki	○		

SmartRead API 全体イメージ

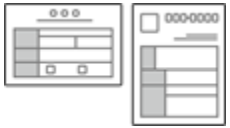


SmartRead API 処理概要

- 基本動作
  - 「タスク」と呼ばれる業務の単位を作成し、その「タスク」に対し、「1リクエスト=1ファイル」で「読取対象文書」をSmartReadにPOSTします。
    - 定型文書を読み取る場合、事前にテンプレートの作成が必要になり、「タスク」にそのテンプレート（テンプレートID）の指定が必要になります。
  - その後、出力処理を実行し、結果をダウンロードします。

- フィールドAPI
  - 「読取対象文書」そのものをPOSTするのではなく、項目画像（文書上に書かれた読取項目だけを画像抽出したもの）をPOSTすることも可能です。
    - その際、使用するAPIのことを「フィールドAPI」と呼びます。
    - 「フィールドAPI」は、「SmartRead API v3」で提供されておらず、「SmartRead API v2」でのみ動作します。
  - その後、読取結果をResponseのJSON形式でGETします。

文書ファイルそのもの

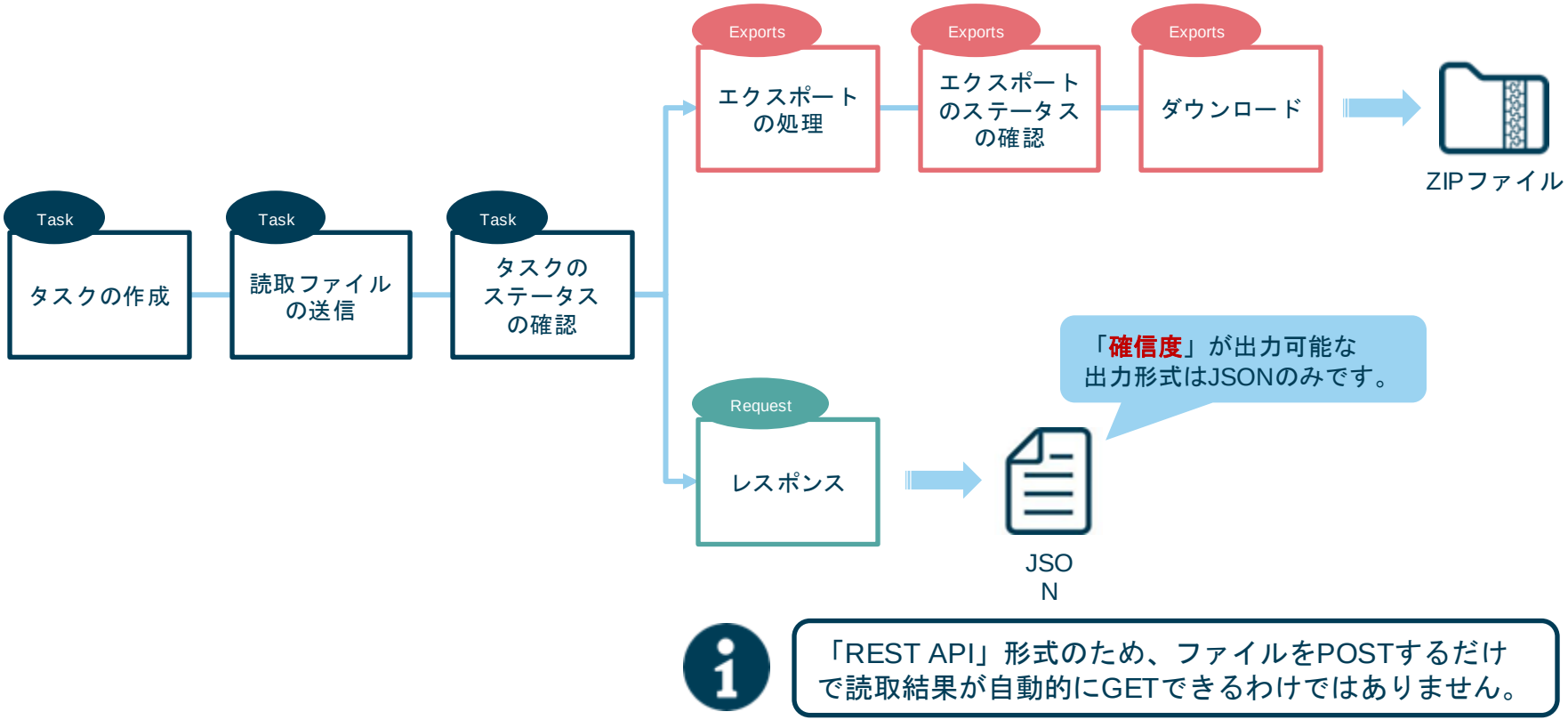


文書上に書かれた読取項目だけを画像抽出したもの

読取対象文書と使用する SmartRead API のバージョン

文書種類	内容	テンプレート	API v2	仕分け（API v3）	API v3
定型文書	1ファイル（1ページ）	必要	○	-	○
	マルチページファイル	必要	×	○	○
非定型文書	預金通帳	不要	×	-	○
	請求書	不要	×	-	○
全文読取	1ファイル（1ページ）	不要	×	-	○
	マルチページファイル	不要	×	-	○
その他	表	不要	×	-	×
	項目画像	不要	○	-	×

読取処理の基本的な SmartRead API v3 処理の流れ



SmartRead API v3 で提供される主な機能 1/2

API	機能	内容
Task	タスクを作成する	空のタスクを作成します。読取処理には最初に必ず作成する必要があります。
	タスクの一覧を取得する	所属するグループのタスクの一覧を取得します。
	タスクの詳細を取得する	タスクの詳細と、処理中のリクエストおよびファイルのサマリーを取得します。 Export API を実行する前に必ず実行する必要があります。
	タスクを更新する	タスクの名前、詳細、ラベルを更新します。 既存のタスク情報を更新したい場合に使用します。
	タスクを削除する	指定したタスクIDのタスクを削除します。
	タスクにファイルを送信する	複数ファイル一括送信はできません。1ファイルずつ送信する必要があります。 ファイルを送信するとリクエストIDが発行されます。
	タスクに含まれるリクエストをリストする	指定したタスクIDに含まれるリクエスト（ファイル）をリストします。
	出力の設定を表示	タスクで設定された出力形式を表示します。
	出力の設定を更新	タスクで設定された出力形式を更新します。 既存のタスク情報を更新したい場合に使用します。

SmartRead API v3 で提供される主な機能 2/2

API	機能	内容
Export	タスクから出力を作成する	処理結果を出力します。 この時点で、処理結果はSmartRead上に存在しています。
	出力の詳細を取得する	Exportのステータスを取得します。
	出力結果をダウンロードする	出力結果をZIPファイル形式でダウンロードします。 ダウンロード後、ZIPファイルの解凍が必要です。
Request	読取ステータスもしくは読取結果を取得	指定したリクエストIDのリクエストの詳細を取得します。
	リクエスト結果を取得する	処理結果をJSON形式で取得します。
Template	テンプレートをSmartReadに登録する	テンプレートを登録します。 登録はWebUIでも可能です。
	テンプレートデータを更新する	テンプレートのデータを更新します。
	テンプレートの詳細を入手する	テンプレートのメタデータとフィールド情報を取得します。
	テンプレートを削除する	指定したテンプレートIDのテンプレートを削除します。

02 事前準備

事前準備

1. APIキーの入手
2. テンプレートの作成（定型文書を読み取る場合のみ）
3. テンプレートIDの入手（定型文書を読み取る場合のみ）
4. APIドキュメントの確認

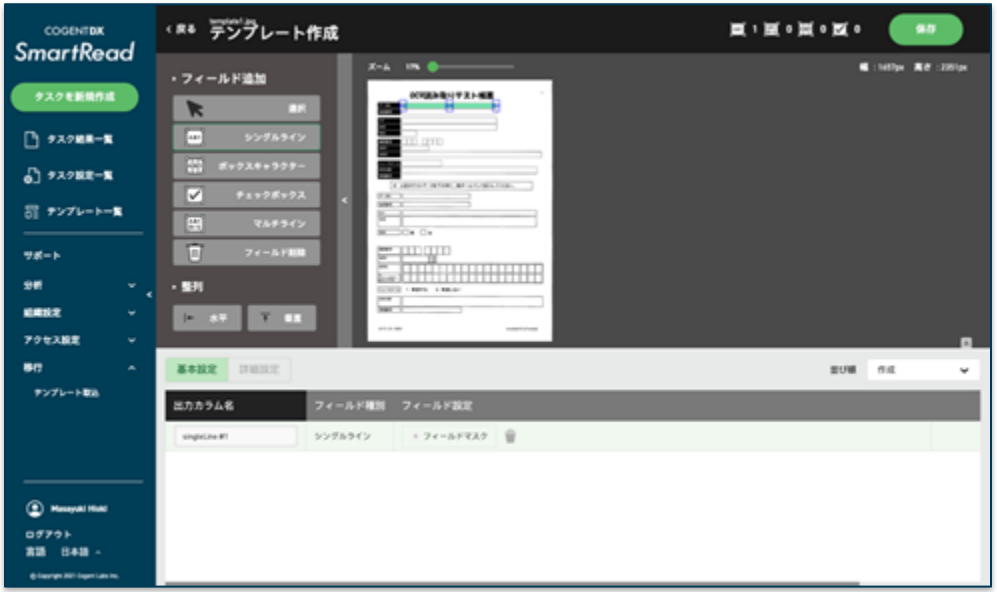
1. APIキーの入手

1. WebUIの「マイページ」ページで確認します。（左メニューの名前をクリック）
2. APIキーをクリックすると、クリップボードにコピーされます。

WebUIを利用しないAPIのみ使用可能な「APIユーザー」も作成可能です。

2. テンプレートの作成（WebUIで作成）

1. WebUI上で作成します。
2. テンプレートは作成後、SmartRead上で保存されます。
（JSONファイルとしてエクスポートすることも可能です。）

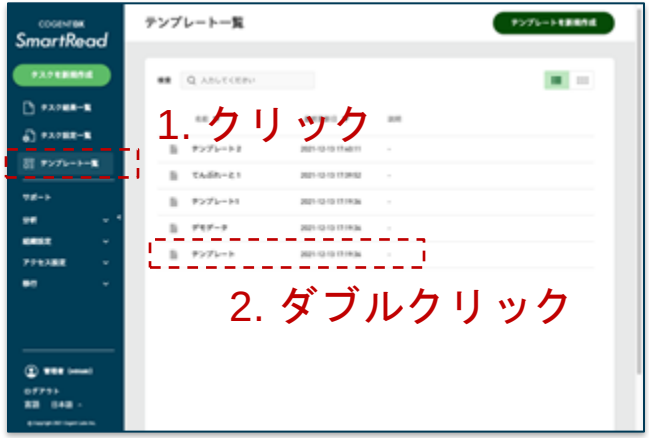


3. テンプレートIDの入手

1. WebUI上でテンプレートを作成します。
2. 該当のテンプレート詳細ページを開きます。
3. Webページ内でテンプレートIDを確認することができません。
4. WebブラウザのURL欄の文字列がテンプレートIDになります。
   こちらのIDをプログラム内で指定してください。

テンプレートID

https://app.smartread.jp/template/XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX



※タスクIDも同様に、「タスク結果詳細」ページのURL欄でタスクIDを確認します。

4. APIドキュメント（https://docs.smartread.jp/）

※SmartRead API v4 用APIドキュメント（https://docs-gsr.smartread.jp/）

APIドキュメント ヘッダーメニュー

ページ	内容
概要	• SmartRead用語の解説 • 基本的なAPIの使い方
APIメソッド	• API機能/仕様/設定方法 • レスポンス内容/形式 • サンプルコード
エラーコード	• HTTPステータスコード • 内部エラーコード
出力フォーマット	• 読取対象文書の種類と出力形式 • JSON出力スキーマ

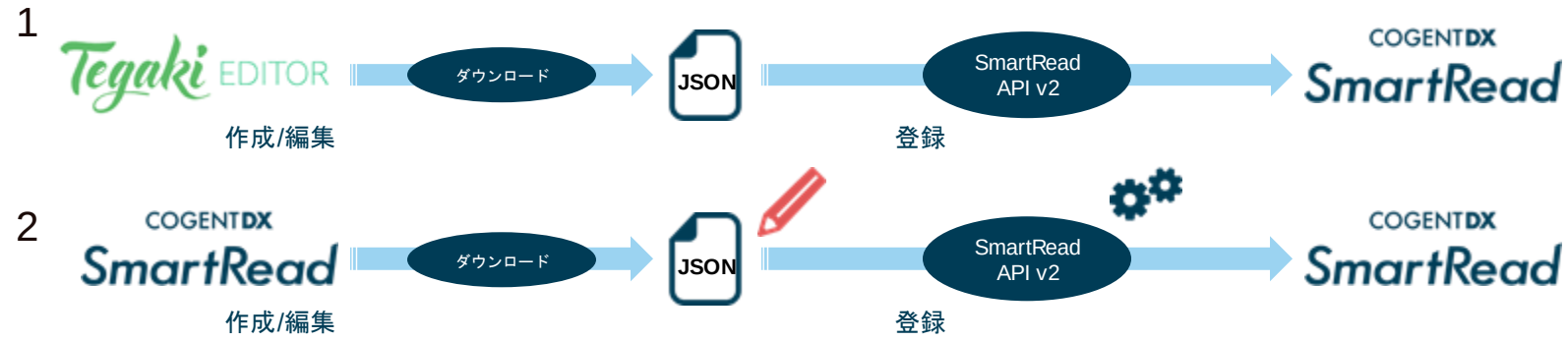
03 SmartRead API v2 使用時の留意点

SmartRead WebUI

SmartRead WebUI は、SmartRead API v3 用に作られています。
このため、WebUI上で SmartRead API v2 に関する以下の操作、ダウンロードを行うことができません。

- テンプレート
- 読取結果結果

SmartRead API v2 用テンプレートの作成/登録/編集



1. Tegaki Editor を使用する場合
   - そのまま SmartRead API v2 でも利用可能です。
   - SmartReadへの登録は SmartRead API v2 を使用してください。
     - SmartRead WebUI での登録はできません。

1. SmartRead WebUI を使用する場合
   - そのままでは SmartRead API v2 で利用できません。
     - JSONファイルを編集（1行追加）してください。
   - SmartReadへの登録は SmartRead API v2 を使用してください。
     - SmartRead WebUI での登録はできません。

```
{
"requestType": "templateMatching",
"templateIds": [
"97dfd517-e13e-4d34-aab0-1ee27ae79b85",
"55e6111f-b20f-4a12-bdf5-0a6c66c055a0"
],
…
```

```
{
 "version": "2",
"requestType": "templateMatching",
"templateIds": [
"97dfd517-e13e-4d34-aab0-1ee27ae79b85",
"55e6111f-b20f-4a12-bdf5-0a6c66c055a0"
],
…
```

Tegaki API v2 からの SmartRead API v2 への移行

Tegaki API v2 は、SmartRead API v2 と同じ仕様です。
このため、既存のTegaki用プログラムはほぼそのままご利用可能ですが、
移行時には以下の留意点をご確認ください。

変更項目	SmartReadで動作させるために必要となる作業
エンドポイント	https://api.tegaki.ai/hwr/v2/... ↓ https://api.**smartread.jp**/hwr/v2/...
APIキー	SmartRead用APIキーの設定が必要 1. SmartRead WebUI 「マイページ」を表示 2. ページ上のAPIキーをクリップボードにコピー 3. 既存プログラム内APIキー→コピーしたAPIキー
テンプレートID	SmartRead用テンプレートIDの設定が必要 1. TegakiからテンプレートJSONファイルをエクスポート 2. SmartRead API v2 を使いJSONファイルを登録（WebUIでの登録は不可） 3. 既存プログラム内テンプレート→ID戻り値であるテンプレートID

04 定型文書の読取処理の実行例（Shell+Curl）

※JSONファイルを読み込む形式での実行例です。
※*斜体文字列*は環境によって入力値が変わります。

エクスポートを使用した読取処理フロー

Step	
1	タスクを作成（POST） → タスクIDを取得
2	タスクIDに対し、ファイルをアップロード（POST） → リクエストIDを取得
3	タスク・ステータスの確認（GET）
4	タスクIDに対し、エクスポートを実行（POST） → エクスポートIDを取得
5	エクスポート・ステータスの確認（GET）
6	タスクID, エクスポートIDに対し、結果を取得（GET） → 結果をZIPファイルで取得

INPUTの定義

リクエスト内容の定義
（例：「預金通帳」を読みたい）

OUTPUTの定義

出力形式の定義
（例：「Excel」で出力したい）

※出力形式はタスク作成時でも定義可能です。

ステータス確認処理は必ず実行してください。
（Step3, 5）

INPUTの定義

タスクの種類	読取対象文書・処理内容
form	定型文書 「仕分処理無（テンプレート1種類）」
bankBook	預金通帳
invoice	請求書
templateMatching	定型文書 「仕分処理から読取処理まで」を自動実行
templateMatchingOnly	定型文書 「仕分処理のみ」を実行
templateMatchingWithVerification	定型文書 「仕分処理まで」を実行
freeform	非定型文書

OUTPUTの定義

出力タイプ	出力結果
csv	CSV
excel	Excel
searchablePdf	テキスト付PDF （Searchable PDF）
json	JSON

Step1：タスクを作成（POST）

- 空タスクの作成

サンプルプログラム

```
#!/bin/bash

curl -X POST ¥
   -H "Authorization:apikey XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" ¥
   -H "Content-Type:application/json" ¥
   --data-binary "@/Work/task.json" ¥
   https://api.smartread.jp/v3/task
```

サンプルJSONファイル（task.json）

```
{
"requestType": ” templateMatching",
"templateIds": [
 ”XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
 ”XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"
],
“name”: ”Sample",
"description": ”Test  TemplateMatching",
"labels": {
 "classification": ”資料掲載用のサンプル読み取りです。"
},
"languages": [
 "ja"
]
}
```

レスポンス結果

```
{"taskId":”XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"}
```

Step2：タスクIDに対し、ファイルをアップロード（POST）

- Step1で作成したタスク（タスクID）へ読取対象文書の送信

サンプルプログラム

```
#!/bin/bash

curl -X POST ¥
  -H "Authorization:apikey XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" ¥
  -F "image=@/Work/payroll.png" ¥
  https://api.smartread.jp/v3/task/:taskId/request
```

Step1で作成されたタスクID

レスポンス結果

```
{"requestId":"XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"}
```

Step3：タスク・ステータスの確認（GET）

- 正常完了の確認（「OCR_COMPLETED」）

サンプルプログラム

```
#!/bin/bash

curl -X GET ¥
  -H "Authorization:apikey XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" ¥
  -H "Content-Type:application/json" ¥
  --data-binary "@/Work/export.json" ¥
  https://api.smartread.jp/v3/task/:taskId
```

Step1で作成されたタスクID

結果

{"userId":"XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX","name":"Payroll","labels":{"classification":"Test"},"languages":["ja"],"description":"Sample test","allowUseOfData":true,"requestType":"payrollReport","modified":"2022-01-13T07:52:05.845Z","created":"2022-01-13T07:52:05.844Z","orgId":"XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX","groupId":"XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX","taskId":"XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX","state":"OCR_RUNNING","requestStateSummary":{"OCR_RUNNING":0,"OCR_COMPLETED":1,"OCR_FAILED":0,"OCR_VERIFICATION_COMPLETED":0,"SORTING_RUNNING":0,"SORTING_COMPLETED":0,"SORTING_FAILED":0},"numOfRequests":1,"numOfPages":1,"formStateSummary":{"OCR_RUNNING":0,"OCR_COMPLETED":1,"OCR_FAILED":0,"OCR_VERIFICATION_COMPLETED":0,"SORTING_RUNNING":0,"SORTING_COMPLETED":0,"SORTING_FAILED":0,"SORTING_DROPPED":0}}

Step5：エクスポート・ステータスの確認（GET）

- 正常完了の確認（「COMPLETED」）

サンプルプログラム

```
#!/bin/bash

curl -X GET ¥
  -H "Authorization:apikey XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" ¥
  -H "Content-Type:application/json" ¥
  --data-binary "@/Work/export.json" ¥
  https://api.smartread.jp/v3/task/:taskId/export/:exportId
```

Step1で作成されたタスクID

Step4で作成されたエクスポートID

レスポンス結果

```
{"type":"excel","aggregation":"oneFile",
"exportId":"XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
"state":"COMPLETED",
"taskId":"XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"}
```

Step6：読取結結果ファイルの取得

- ZIPファイルの取得

サンプルプログラム

```
#!/bin/bash

curl -X GET ¥
  -H "Authorization:apikey XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" ¥
  -o " @/ Work/v3_Task/result.zip" ¥
  https://api.smartread.jp/v3/task/:taskId/export/:exportId/download
```

Step1で作成されたタスクID

Step4で作成されたエクスポートID

レスポンス結果

```
result.zip
```

Step4：タスクIDに対し、エクスポートを実行（POST）

- エクスポート処理の実行（CSV出力の場合）

サンプルプログラム

```
#!/bin/bash

curl -X POST ¥
  -H "Authorization:apikey XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" ¥
  -H "Content-Type:application/json" ¥
  --data-binary "@/Work/export.json" ¥
  https://api.smartread.jp/v3/task/:taskId/export
```

Step1で作成されたタスクID

サンプルJSONファイル（export.json）

```
{
"type": "csv",
"aggregation": "oneFile"
}
```

結果

```
{"exportId":"XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"}
```

05 読取処理の自動化

SmartReadと他社製品との連携

APIを使うことで、他システム連携/自動化を実現することができます。

プログラム開発の敷居を下げるため、内部的なAPIの動きを意識せずとも、
容易にシステム連携できる連携ソリューションをご用意しています。

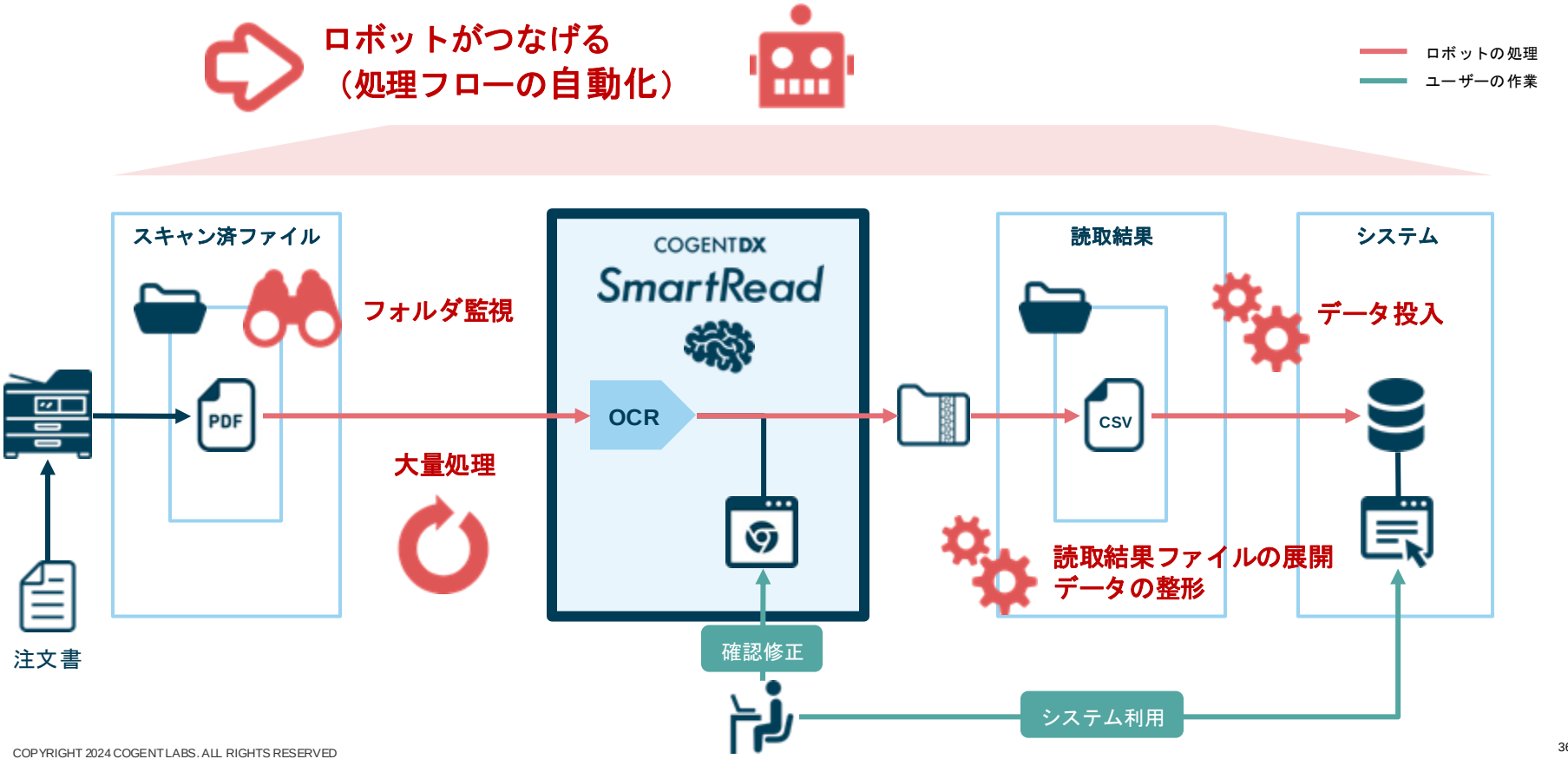
- 他社製品標準機能への組み込み
- 連携コネクター
- サンプルプログラム

RPA

BizteX Connect

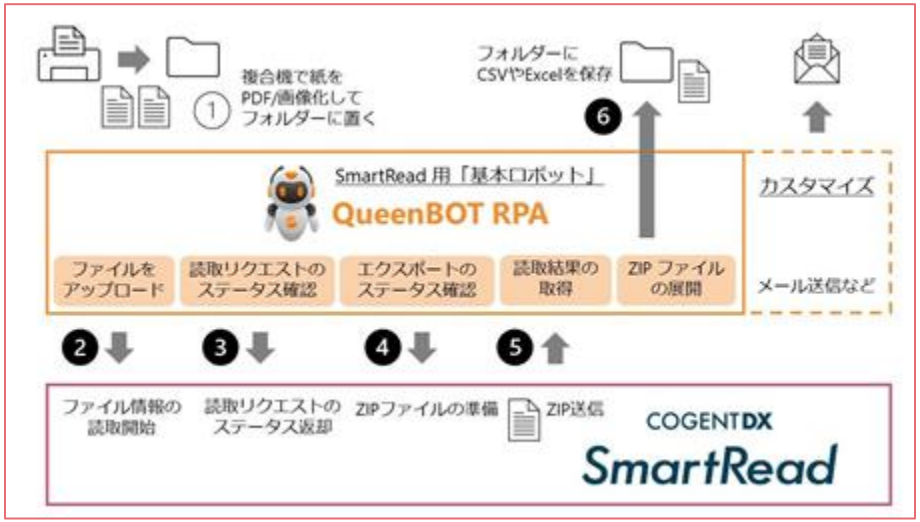
iPaaS

他ソリューションとのAPI連携による読取処理の自動化イメージ



RPAとの連携

読取処理の基本ステップをパラメータ設定のみですぐに業務フローに
組み込めるRPA用ロボットパーツ「RPAコネクター」を無償提供しています。



QueenBOT RPA の場合

- BizRobo!
- Blue Prism
- QueenBOT RPA
- UiPath
- WinActor

「Power Automate」用ロボットフローについては、
サンプルプログラム、もしくは
パートナー様ソリューションとしてご提供いたします。

RPAコネクターの中身

```
SmartReadSample
├サンプルロボット                    … サンプルロボット格納フォルダ
│ ├サンプルロボット.ums7              … サンプルロボット本体
├ライブラリ                          … コネクター格納フォルダ
│ ├エクスポート.ums7                 … エクスポート API 用
│ ├エクスポート確認.ums7             … エクスポート確認 API 用
│ ├ダウンロード.ums7                 … ダウンロード API 用
│ ├タスク作成.ums7                   … タスク作成 API 用
│ ├リクエスト.ums7                   … リクエスト API 用
│ ├リクエスト確認.ums7l              … リクエスト確認 API 用
│ ├削除.ums7                         … 削除 API 用
├input                               … API 用 JSON 格納フォルダ
│ ├sample                           … サンプル JSON 格納フォルダ
│ │ ├export_csv.json                … CSV エクスポート用 Sample（定型文書用）
│ │ ├export_excel.json              … Excel エクスポート用 Sample（定型文書用）
│ │ ├export_pdf.json                … PDF エクスポート用 Sample（主に非定型用）
│ │ ├task_freeform.json             … 非定型文書のタスク用 Sample
│ │ ├task_verification.json         … 定型文書の仕分け・読取確認有りタスク用 Sample
│ │ ├task_withoutverification.json  … 定型文書の仕分け・読取確認無しタスク用 Sample
│ ├export.json                      … エクスポート用 JSON
│ └task.json                        … タスク作成用 JSON
├download                           … ダウンロードファイル格納フォルダ
├upload                             … アップロードファイル格納フォルダ
└response                           … レスポンス JSON ファイル格納フォルダ
```

※WinActorコネクターの場合

- マニュアル
- 各種ロボットと関連フォルダ
- 基本動作サンプルプログラム

SmartRead API v3 を使用しています。

Power Automate for Deskotp サンプルプログラム

- 定型文書を読み取り、読取結果をExcel形式で出力
- 定型文書を読み取り、読取結果をkintoneにインポート
- PDFを読み取り、テキスト付PDF形式で出力

Biztex Connect (iPaaS) との連携

標準でSmartReadが利用可能です。

メール、クラウドサービスとの連携をWeb画面上で容易に作成できます。

BizteX Connect

参考 用語の解説

押さえておくべき主な用語

- APIキー
- テンプレート
- タスク
- 仕分け

APIキー

SmartRead API を実行する際に必要となる認証キーになります。
APIキーは、SmartRead WebUI 上で確認します。

```
headers = {"Authorization": "apikey  XXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"}
```

APIキー

テンプレート（定型文書読取時のみ必要）

テンプレート

雛形　フォーム　フォーマット のようなもので

記入箇所が定義された設定ファイル のこと

※テンプレートにより、どの位置（箇所）にどういう内容が記載されているか定義が可能

タスク

・どのテンプレートを使う？
・結果はExcel出力する？
…

読取処理の受付単位（≒業務）

※複数テンプレートを設定した場合、SmartReadが自動的に仕分処理

SmartRead API v3 を使用する場合には、
必ず「タスク」の作成が必要になります。

仕分け（仕分処理）

読み取った定型文書が
どのテンプレートで作られた文書か？
を仕分けするSmartRead上の処理機能のこと

✕読取対象文書ファイルを種類毎に
Windows OS 上でフォルダ分類する

免責事項

- 本文書はご参考資料としてご提供するものです。本文書に関して弊社は質問へのご回答などサポートのご提供はいたしません。
- 本文書は作成時において弊社が知り得た情報をもとに作成したものです。弊社は、本文書の正確性、網羅性、最新性などのいかなる事項についても保証いたしません。
- 弊社は、本文書に関して、請求の原因の如何を問わず損害賠償責任その他の責任を負いません。