ファイル受け取りシステム 技術仕様書

目次

[１ 概要 3](#_Toc147233099)

[２ 動作環境 4](#_Toc147233100)

[３ 使用方法 5](#_Toc147233101)

[３-１ ファイルを受けとる方法 5](#_Toc147233102)

[３-２ ファイルを送信する方法 7](#_Toc147233103)

[４ 詳細な仕様 8](#_Toc147233104)

[５ ソースコードの確認方法 9](#_Toc147233105)

[６ 構成 10](#_Toc147233106)

[６-１ コンピューターの構成 10](#_Toc147233107)

[６-２ フレームワークやライブラリ等の構成 11](#_Toc147233108)

[６-２-１ 送信プログラムのフレームワークやライブラリ 11](#_Toc147233109)

[６-２-２ 中継プログラムのフレームワークやライブラリ 11](#_Toc147233110)

[６-２-３ 受信プログラムのフレームワークやライブラリ 11](#_Toc147233111)

[６-３ 本番環境におけるプログラムの配置場所 12](#_Toc147233112)

[６-３-１ 送信プログラムの配置場所 12](#_Toc147233113)

[６-３-２ 中継プログラムの配置場所 13](#_Toc147233114)

[６-３-３ 受信プログラムの配置場所 14](#_Toc147233115)

[６-４ ファイル構成 15](#_Toc147233116)

[６-４-１ 送信プログラムのファイル構成 15](#_Toc147233117)

[６-４-２ 中継プログラムのファイル構成 15](#_Toc147233118)

[６-４-３ 受信プログラムのファイル構成 16](#_Toc147233119)

[６-５ 関数一覧 17](#_Toc147233120)

[６-５-１ 送信プログラムの関数一覧 17](#_Toc147233121)

[６-５-２ 中継プログラムの関数一覧 18](#_Toc147233122)

[６-５-３ 受信プログラムの関数一覧 19](#_Toc147233123)

[７ 動作の流れ 20](#_Toc147233124)

[７-１ 大まかな動作の流れ 20](#_Toc147233125)

[７-２ 詳細な動作の流れ 20](#_Toc147233126)

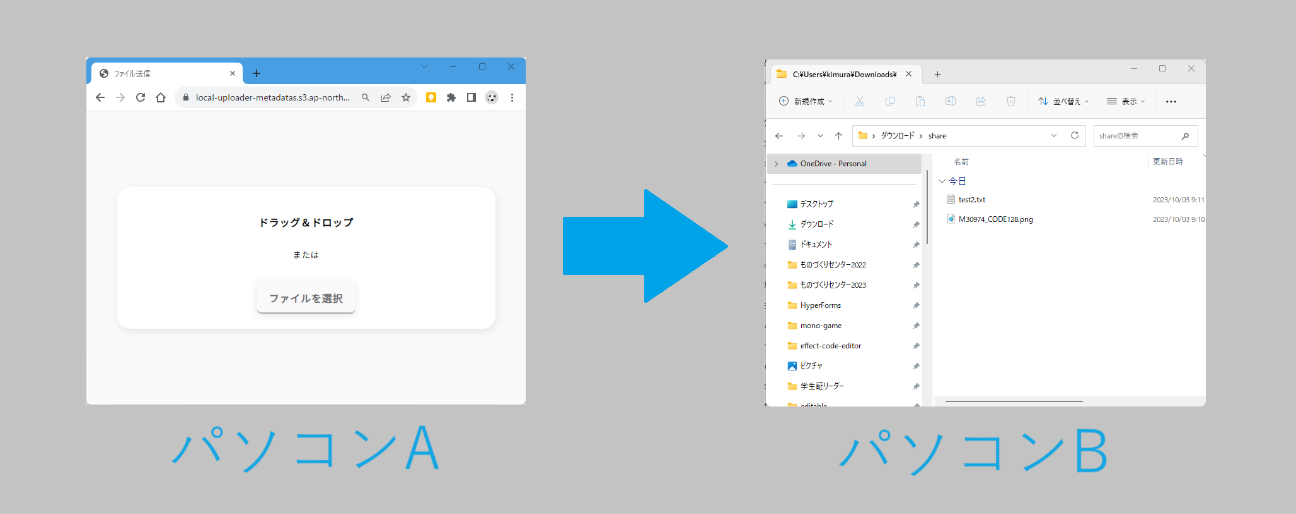
[８ デバッグの方法 21](#_Toc147233127)

[９ 更新方法 22](#_Toc147233128)

# 概要

本システムは、webページから特定のコンピューターへファイルを送信するシステムです。

想定している使用用途は、ものづくりセンター内のプラズマ加工機や３Dプリンター等の機械設備の運用です。不特定多数が機械設備を利用するにあたって、各々が設計したファイルを施設内のwindowsパソコンへ転送する必要があります。このシステムを活用すると、ホームページからファイルを転送することができます。



# 動作環境

本システムを利用するためには、以下の要件を満たすコンピューターが必要です。

**受信側に必要な動作環境**

* Windows 64bitのみ動作確認済。
* インターネットに接続されている必要があります。

**送信側に必要な動作環境**

* ブラウザが動くコンピューターなら、何でもOKです。

# 使用方法

## ファイルを受けとる方法

* **【手順１】**以下のURLまたはQRコードから、受信用アプリケーションをダウンロードしてください。

<https://local-uploader-metadatas.s3.ap-northeast-1.amazonaws.com/download.html>

QR コード

自動的に生成された説明

以下の画像のようにGoogle Chromeにブロックされた場合は、右側の小さい矢印をクリックして、ダウンロードを続行してください。

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明

* **【手順２】**windowsのコンピューターで、受信用アプリケーションを起動してください。「PCが保護されました」と表示された場合は、「詳細情報」をクリックした後、「続行」をクリックしてください。

グラフィカル ユーザー インターフェイス, アプリケーション

自動的に生成された説明　 グラフィカル ユーザー インターフェイス, アプリケーション

自動的に生成された説明

* **【手順３】**受信用アプリケーションが起動します。

**グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明**

* **【手順４】**ファイルを受け取りたいフォルダを選択してください。

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション, メール

自動的に生成された説明

* **【手順５】**送信用webページのURLを、ファイルを送信して欲しい人（学生等）に公開してください。

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明

* **【手順６】**ファイルを受け取る準備ができました。以下の条件を満たしている間、継続的にファイルを受信して、保存先フォルダの中身を更新します。

**条件**

* + 受信用アプリケーションが起動している必要があります。
  + 受信側のコンピューターがインターネットに接続されている必要があります。
  + 受け取り用のフォルダが存在している必要があります。

## ファイルを送信する方法

受信側の担当者からURLを教えて貰い、アクセスして、ファイルをアップロードしてください。

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション, PowerPoint

自動的に生成された説明

# 詳細な仕様

* 受信側のファイルの更新頻度は、15秒につき1回です。アップロードした後に15秒以内に反映されない場合は、何らかの不具合が発生しています。
* 受信側のパソコンの電源が切れている場合もアップロードは可能です。受信側のパソコンの電源を入れた際に、電源が落ちていた間に受信したファイルを、まとめてダウンロードします。
* １台のコンピューターの中で、受信用アプリケーションを２つ以上同時に動かすことはできません。
* 送信用のwebページのURLは、受信側のデバイスごとに固有です。
  + 受信用アプリケーションを再起動した場合 → URLは変わりません。
  + 受信用パソコンの中身の回路基板を交換した場合 → URLが変わる可能性があります。
  + 受信するフォルダを変えた場合 → URLは変わりません。

# ソースコードの確認方法

ソースコードの全文は、以下のGitHubリポジトリに保管しています。

<https://github.com/bibakichi/share_file_to_local>

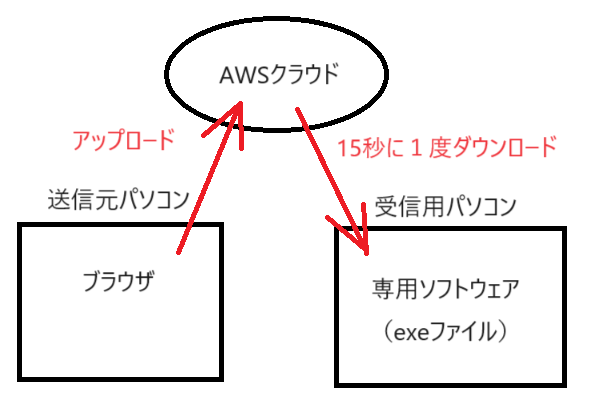
ただし、GitHub上のコードを書き換えても本番環境に反映されません。CI/CD（※）等の設定は行っていないためです。更新の際は、各所に散らばっている本番環境のプログラムを、手動で入れ換える必要があります。

※CI/CDとは、GitHub上のコードを自動で本番環境に反映させる仕組みのこと。

# 構成

## コンピューターの構成

本システムの構成を説明するうえで、２台のコンピューターと１つのクラウドが登場します。それぞれ送信元パソコンと受信用パソコン、そして中継用のAWSクラウドです。



## フレームワークやライブラリ等の構成

### 送信プログラムのフレームワークやライブラリ

送信プログラムはHTML, CSS, JavaScriptで書かれており、ブラウザ上で動作します。つまり、送信側のプログラムは単なるwebページだと思ってください。jQueryなどのライブラリやフレームワークは使っていません。

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション, PowerPoint

自動的に生成された説明

### 中継プログラムのフレームワークやライブラリ

AWS S3とLambdaで構成しています。Lambdaの言語はnode.jsです。npmパッケージはデフォルトのものしか使用しておらず、追加のインストールは不要です。データベースやAWS API Gateway、AWS CloudFrontは使っておりません。

### 受信プログラムのフレームワークやライブラリ

Node.jsとelectronを使っています。また、パッケージ管理にはnpmを使用しています。ビルドにはいくつかのパッケージが必要です。詳しくは ”package.json” を確認してください。

**グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明**

## 本番環境におけるプログラムの配置場所

### 送信プログラムの配置場所

送信元パソコンで動作するwebページは、AWS S3に静的ファイルとして配置しています。配置場所は、バケット「local-uploader-metadatas」の直下です。専用のバケットを用意するのが勿体なかったため、メタデータ用のバケットに入れています。ファイルは ”index.html” の１つのみです。（CSSとJavaScriptをインラインで記述しているため）

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション, PowerPoint

自動的に生成された説明

AWS S3の管理画面には、以下のURLからアクセスできます。

<https://s3.console.aws.amazon.com/s3/buckets/local-uploader-metadatas?region=ap-northeast-1>

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明

### 中継プログラムの配置場所

中継プログラムは、AWSのクラウド上に展開しています。

### 受信プログラムの配置場所

受信プログラムは、実行可能形式（exeファイル）で事務室内のパソコンにインストールしています。

## ファイル構成

### 送信プログラムのファイル構成

ファイルは ”index.html” の１つのみです。（CSSとJavaScriptをインラインで記述しているため）

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション, PowerPoint

自動的に生成された説明

### 中継プログラムのファイル構成

* Lambda関数「local\_uploader\_sign\_url」
  + ***index.mjs*** …
* Lambda関数「local\_uploader\_handle\_change」
  + ***index.mjs*** …
* S3バケット「local-uploader-files」

→受け取ったファイルを保存するためのバケットです。

* S3バケット「local-uploader-metadatas」

→メタデータを配信するためのバケットです。

### 受信プログラムのファイル構成

* ***forge.config.js*** … 実行可能ファイルに変換するための設定を記述しています。詳細は「Electron Forge」のホームページを参照してください。<https://www.electronforge.io/>
* ***package.json*** …
* ***package-lock.json*** …
* フォルダ「src」
  + ***main.js*** …
  + ***thread\_main.js*** …
  + ***thread\_sub.js*** …
  + ***index.html*** …
  + ***style.css*** …
  + ***preload.js*** …
  + ***get\_target\_folder.js*** …
  + ***get\_device\_id.js*** …

## 関数一覧

### 送信プログラムの関数一覧

* ファイル「index.html」
  + ***関数「window.onload」*** … ページが読み込まれた直後に実行する関数。入力フォームやドラッグ＆ドロップのエリアの初期設定を行う。
  + ***関数「uploadMultipleFiles」*** … 複数のファイルを一斉送信する関数。
  + ***関数「uploadFile」*** … ファイル１つを送信する関数。
  + ***関数「showLoader」*** … ローダー（アップロード中のグルグル）を表示する関数。
  + ***関数「deleteLoader」*** … ローダー（アップロード中のグルグル）を消す関数。

### 中継プログラムの関数一覧

* Lambda関数「local\_uploader\_sign\_url」

このLambda関数は、ユーザーがファイルをアップロードする直前に起動します。

* + ファイル「index.mjs」
    - ***メイン関数*** … このLambda関数は、メイン関数のみで動作しています。
* Lambda関数「local\_uploader\_handle\_change」

このLambda関数は、S3にファイルがアップロードされたときに起動します。

* + ファイル「index.mjs」
    - ***メイン関数*** … アップロードされた全てのファイルについて処理を行います。for文で繰り返して、以下の関数「\_oneRecord」を呼び出します。
    - ***関数「\_oneRecord」***… アップロードされたファイルを１個ずつ処理します。
    - ***関数「\_streamToString」***… テキストファイルのデータを、文字列に変換します。

### 受信プログラムの関数一覧

* フォルダ「src」
  + ファイル「main.js」
    - ***メイン処理*** … ２つのスレッドを起動させる。２つのスレッドは同時並列に実行される。
  + ファイル「thread\_main.js」
    - ***メイン関数*** … メインスレッド。GUIの処理を担当する。
    - ***関数「createWindow」*** … GUIの操作画面を開く関数。
  + ファイル「thread\_sub.js」
    - ***メイン関数*** … サブスレッド。ファイルの受信処理を担当する。バックグラウンド処理として働き、15秒に１度サーバーを監視する。
    - ***関数「\_update」*** … サーバーを監視して、ファイルが追加されていたらダウンロードする。
    - ***関数「\_getOldMetaData」*** … ローカルコンピューターに保存されている、古いメタデータを取得する。
    - ***関数「\_getNewMetaData」*** … サーバーで管理されている、最新のメタデータを取得する。
    - ***関数「\_saveFile」*** … 任意のURLからバイナリファイルを取得して、ファイルとして保存する関数。
  + ファイル「preload.js」
    - ***イベント「DOMContentLoaded」*** … ページが読み込まれた直後に実行されるイベント。
    - ***イベント「selected-directory」*** … メインプロセスから送られたデータを受信したときに、実行されるイベント。
    - ***イベント「render-my-data」*** … メインプロセスから送られたデータを受信したときに、実行されるイベント。
  + ファイル「get\_target\_folder.js」
    - ***メイン関数*** … どこに保存すればいいかを取得する関数。
  + ファイル「get\_device\_id.js」
    - ***メイン関数*** … デバイスごとに固有のIDを取得する関数。

# 動作の流れ

## 大まかな動作の流れ

## 詳細な動作の流れ

# デバッグの方法

C:\Users\kimura\AppData\Roaming\share\_file\_client\logs

# 更新方法

GitHubからコードを自動送信するような、CI/CD等の設定は行っておりません。更新の際は、コードを手動で書き換える必要があります。