aws

ウェブクライアントSDKデベロッパーガイド

Amazon DCV

Amazon DCV: ウェブクライアントSDKデベロッパーガイド

Table of Contents

Amazon DCV Web Client とは SDK

> **Note**
>
> Amazon DCVは以前NICEは と呼ばれていましたDCV。

Amazon DCVは高性能のリモートディスプレイプロトコルです。さまざまなネットワーク条件で、リモートデスクトップやアプリケーションストリーミングをクラウドやデータセンターからあらゆるデバイスへ安全に配信できます。Amazon DCVで Amazon を使用するとEC2、Amazon EC2インスタンスでグラフィックを多用するアプリケーションをリモートで実行できます。結果をより控えめなクライアントマシンにストリーミングできるため、高価な専用ワークステーションが不要になります。

Amazon DCV Web Client SDKは、独自の Amazon DCV ウェブブラウザクライアントアプリケーションを開発するために使用できる JavaScript ライブラリです。エンドユーザーは、これらのアプリケーションを使用して、実行中の Amazon DCVセッションに接続して操作できます。

Amazon DCV Web Client をビルディングブロックSDKとして使用すると、カスタマイズされたウェブアプリケーションを構築できます。これにより、ユーザーはどこからでもデスクトップやアプリケーションに瞬時にアクセスでき、ネイティブにインストールされたアプリケーションとほとんど区別できない応答性と流体パフォーマンスが得られます。

このガイドでは、Amazon DCV Web Client SDK を使用してカスタムウェブブラウザクライアントアプリケーションを構築し、ワークフロー内で Amazon DCVセッションとやり取りする方法について説明します。

トピック

前提条件

Amazon DCV Web Client の使用を開始する前にSDK、Amazon DCVおよび Amazon DCVセッションに精通していることを確認してください。詳細については、「Amazon DCV管理者ガイド」を参照してください。

Amazon DCV Web Client は、Amazon DCVサーバーバージョン 2020 以降SDKをサポートしています。

サポートされている機能

以下の Amazon DCV機能をサポートするカスタムウェブブラウザクライアントアプリケーションを構築できます。

- Windows Amazon DCVサーバーに接続する
- Linux Amazon DCVサーバーに接続する
- ストリーミングモードの管理
- ファイルの転送
- セッションから印刷
- コピーアンドペースト
- ステレオ 2.0 オーディオ再生
- ステレオ 2.0 オーディオ録音 (Windows サーバーの場合)
- タッチスクリーン
- スタイラス (Linux、Windows 10、Windows Server 2019 の各サーバー)
- マルチモニターのサポート

これらの機能の詳細については、「Amazon DCVユーザーガイド」の「サポートされている機能」を参照してください。

ブラウザのサポート

Amazon DCV Web Client は JavaScript （ ES6) SDKをサポートしており、 JavaScript または TypeScript アプリケーションから使用できます。

Amazon DCV Web Client は、次のウェブブラウザSDKをサポートしています。

ブラウザ	Version
Google Chrome	最新の 3 つのメジャーバージョン
Mozilla Firefox	最新の 3 つのメジャーバージョン
Microsoft Edge	最新の 3 つのメジャーバージョン
MacOS 版 Apple Safari	最新の 3 つのメジャーバージョン

バージョニング規則

Amazon DCV Web Client SDKのバージョンは、 の形式で定義されます*major*.*minor*.*patch*。バージョニング規則は、通常、セマンティックバージョニングモデルに準拠しています。メジャーバージョンの変更 (`1.x.x` から `2.x.x` への変更など) は、互換性を破る変更が導入され、それによってコードの変更や計画されたデプロイが必要になる可能性があることを示します。マイナーバージョンの変更 (`1.1.x` から `1.2.x` への変更など) については、下位互換性がありますが、非推奨の要素が含まれている可能性があります。

Amazon DCV Web Client の開始方法 SDK

Amazon DCV Web Client は、メインdcv.jsファイルといくつかの補助コンポーネントSDKで構成されます。すべてのファイルは圧縮アーカイブ内に分散され、Amazon DCVウェブサイト からダウンロードできます。

Amazon DCV Web Client の使用を開始するには SDK

1. Amazon DCV Web Client SDKアーカイブは、安全な署名でデジタルGPG署名されます。アーカイブの署名を確認するには、 NICE GPG キーをインポートする必要があります。これを行うには、ターミナルウィンドウを開き、NICEGPGキーをインポートします。

   ```
   $ wget https://d1uj6qtbmh3dt5.cloudfront.net/NICE-GPG-KEY
   ```

   ```
   $ gpg --import NICE-GPG-KEY
   ```

2. Amazon DCV Web Client SDKアーカイブと Amazon DCV Web Client SDKアーカイブ署名を Amazon DCVウェブサイト からダウンロードします。
3. 署名を使用して Amazon DCV Web Client SDKアーカイブの署名を確認します。

   ```
   $ gpg --verify
   signature_filename.zip.sign
   archive_filename.zip
   ```

   例:

   ```
   $ gpg --verify nice-dcv-web-client-sdk-1.8.4-840.zip.sign nice-dcv-web-client-
   sdk-1.8.4-840.zip
   ```

4. 署名が正常に検証された場合は、Amazon DCV Web Client SDKアーカイブの内容を抽出し、抽出されたディレクトリをウェブサーバーに配置します。例:

   ```
   $ unzip
   archive_filename.zip
   -d /
   path_to
   ```

```
/
server_directory
/
```

> **Important**
>
> - Amazon DCV Web Client をウェブサーバーSDKにデプロイするときは、フォルダ構造を保持する必要があります。
> - Amazon DCV Web UI を使用する場合SDK、React `DCVViewer` コンポーネントは、このパッケージの EULA.txt および third-party-licenses.txt ファイルが埋め込みウェブサーバーのURLパスに存在することを期待します。 third-party-licenses.txt ファイルを変更して、Amazon DCV Web Client SDKパッケージの対応するファイルの内容と、場合によっては消費するユーザーアプリケーションで使用されるライブラリからのその他のライセンス情報も含める必要があります。

Amazon DCVサーバーに接続して最初のフレームを取得する

次のチュートリアルでは、カスタムウェブクライアント用にHTMLページを準備する方法、Amazon DCVサーバーを認証して接続する方法、および Amazon DCVセッションからストリーミングされたコンテンツの最初のフレームを受信する方法を示します。

トピック

ステップ 1: HTMLページを準備する

ウェブページでは、必要な JavaScript モジュールをロードし、Amazon DCV Web Client がリモート Amazon DCVサーバーからコンテンツストリームSDKを描画するid有効な `<div>` HTML 要素を追加する必要があります。

例:

```
<!DOCTYPE html>
```

```
<html lang="en" style="height: 100%;">
  <head>
    <title>DCV first connection</title>
  </head>
  <body style="height: 100%;">
    <div id="root" style="height: 100%;"></div>
    <div id="dcv-display"></div>
    <script type="module" src="index.js"></script>
  </body>
</html>
```

ステップ 2: 最初のフレームの認証、接続、取得を行う

このセクションでは、ユーザー認証プロセスを完了する方法、Amazon DCVサーバーを接続する方法、Amazon DCVサーバーからコンテンツの最初のフレームを受信する方法を示します。

まず、`index.js`ファイルから Amazon DCV Web Client をインポートしますSDK。これは、次のようなユニバーサルモジュール定義 (UMD) モジュールとしてインポートできます。

```
import "./dcvjs/dcv.js"
```

それ以外の`1.1.0`場合は、バージョン から、次のように対応するパッケージからECMAScriptモジュール (ESM) としてインポートすることもできます。

```
import dcv from "./dcvjs/dcv.js"
```

Authentication オブジェクト、Connection オブジェクト、および Amazon DCVサーバーの保存に使用する変数を定義しますURL。

```
let auth,
    connection,
    serverUrl;
```

スクリプトのロード時に Amazon DCV Web Client SDKのバージョンをログに記録し、ページロード時に `main`関数を呼び出します。

```
console.log("Using Amazon DCV Web Client SDK version " + dcv.version.versionStr);
document.addEventListener('DOMContentLoaded', main);
```

main 関数により、ログレベルの設定と認証プロセスの開始が実行されます。

```
function main () {
  console.log("Setting log level to INFO");
  dcv.setLogLevel(dcv.LogLevel.INFO);

  serverUrl = "https://your-dcv-server-url:port/";

  console.log("Starting authentication with", serverUrl);

  auth = dcv.authenticate(
    serverUrl,
    {
      promptCredentials: onPromptCredentials,
      error: onError,
      success: onSuccess
    }
  );
}
```

promptCredentials 関数、error 関数、success 関数は、必須コールバック関数であり、認証プロセスで定義する必要があります。。

Amazon DCVサーバーが認証情報の入力を求められた場合、promptCredentialsコールバック関数は Amazon DCVサーバーから要求された認証情報チャレンジを受け取ります。Amazon DCVサーバーがシステム認証を使用するように設定されている場合は、サインイン認証情報を指定する必要があります。次のコードサンプルでは、仮に、ユーザー名を my_dcv_user 、パスワード my_password にしています。

認証に失敗すると、errorコールバック関数は Amazon DCVサーバーからエラーオブジェクトを受け取ります。

認証が成功すると、successコールバック関数は、my_dcv_userユーザーが Amazon DCVサーバーで接続することを許可されている各セッションのセッション ID (sessionId) と認証トークン (authToken) を含むカップルの配列を受け取ります。次のコードサンプルでは、接続関数を呼び出して、配列で返された最初のセッションに接続します。

> **Note**
>
> 以下のコード例では、MY_DCV_USER を自分のユーザー名に、MY_PASSWORD を自分のパスワードに置き換えてください。

```
function onPromptCredentials(auth, challenge) {
  // Let's check if in challege we have a username and password request
  if (challengeHasField(challenge, "username") && challengeHasField(challenge,
 "password")) {
    auth.sendCredentials({username: MY_DCV_USER, password: MY_PASSWORD})
  } else {
    // Challenge is requesting something else...
  }
}

function challengeHasField(challenge, field) {
  return challenge.requiredCredentials.some(credential => credential.name === field);
}

function onError(auth, error) {
  console.log("Error during the authentication: " + error.message);
}

// We connect to the first session returned
function onSuccess(auth, result) {
  let {sessionId, authToken} = {...result[0]};

  connect(sessionId, authToken);
}
```

Amazon DCVサーバーに接続します。`firstFrame` コールバックメソッドは、最初のフレームがAmazon DCVサーバーから受信されたときに呼び出されます。

```
function connect (sessionId, authToken) {
  console.log(sessionId, authToken);

  dcv.connect({
    url: serverUrl,
    sessionId: sessionId,
    authToken: authToken,
    divId: "dcv-display",
    callbacks: {
      firstFrame: () => console.log("First frame received")
    }
  }).then(function (conn) {
    console.log("Connection established!");
    connection= conn;
  }).catch(function (error) {
```

```
    console.log("Connection failed with error " + error.message);
  });
}
```

ボーナス: HTML ログインフォームを自動的に作成する

challenge オブジェクトは promptCredentials コールバック関数が呼び出されたときに返されます。これには、オブジェクトの配列requiredCredentialsである という名前のプロパティが含まれます。これは、Amazon DCVサーバーによってリクエストされる認証情報ごとに 1 つのオブジェクトです。各オブジェクトには、リクエストされた認証情報の名前とタイプが含まれます。challenge および requiredCredentials オブジェクトを使用して、HTMLログインフォームを自動的に作成できます。

次のコードサンプルでは、その実行方法を説明しています。

```
let form,
    fieldSet;

function submitCredentials (e) {
  var credentials = {};
  fieldSet.childNodes.forEach(input => credentials[input.id] = input.value);
  auth.sendCredentials(credentials);
  e.preventDefault();
}

function createLoginForm () {
  var submitButton = document.createElement("button");

  submitButton.type = "submit";
  submitButton.textContent = "Login";

  form = document.createElement("form");
  fieldSet = document.createElement("fieldset");

  form.onsubmit = submitCredentials;
  form.appendChild(fieldSet);
  form.appendChild(submitButton);

  document.body.appendChild(form);
}

function addInput (name) {
```

```
  var type = name === "password" ? "password" : "text";

  var inputField = document.createElement("input");
  inputField.name = name;
  inputField.id = name;
  inputField.placeholder = name;
  inputField.type = type;
  fieldSet.appendChild(inputField);
}

function onPromptCredentials (_, credentialsChallenge) {
  createLoginForm();
  credentialsChallenge.requiredCredentials.forEach(challenge =>
 addInput(challenge.name));
}
```

Amazon DCVの機能を使用する

Amazon DCV機能の可用性は、Amazon DCVセッション用に設定されたアクセス許可とクライアントのウェブブラウザの機能によって異なります。

Amazon DCVセッションで使用できる機能は、セッションに指定されたアクセス許可によって管理されます。つまり、Amazon DCV Web Client で機能がサポートされている場合でもSDK、セッション管理者が定義したアクセス許可に基づいて、その機能へのアクセスが妨げられる可能性があります。詳細については、「Amazon 管理者ガイド」の「Amazon DCV認証の設定」を参照してください。 DCV

featuresUpdate コールバック関数について

Amazon DCVセッションの機能の可用性が変更されると、Amazon DCV Web Client は接続の確立時に指定したfeaturesUpdateコールバック関数を使用してSDK通知します。例:

```
featuresUpdate: function (connection, list) {
  ...
},
```

コールバック関数は、可用性が変化した機能のみを通知します。list パラメータは一連の文字列で、更新された機能の名前のみが含まれます。例えば、セッションでオーディオ入力機能の可用性が変化した場合、パラメータには ["audio-in"] のみが含まれます。後に、セッションでクリッ

プボードのコピー/貼り付け機能の可用性が変わった場合は、パラメータに ["clipboard-copy", "clipboard-paste"] のみが含まれます。

機能更新の処理

featuresUpdate コールバック関数は、可用性が変化した機能のみを通知します。どの機能が更新されたかを知るには、connection.queryFeature メソッドを使用して機能のクエリを行う必要があります。これは、変更通知の受信後にいつでも実行できます。このメソッドでは、解決を行う Promise が、リクエストされた機能の更新ステータスに返されます。status 値は常に関連付けられ、enabled というブール値 (true|false) プロパティを含みます。場合によっては、status 値に追加のプロパティがある機能もあります。機能の可用性が更新されていない場合は、拒否されます。

次のコード例では、その実行方法を説明しています。

```
// Connection callback called
function featuresUpdate (_, list) {
  if (list.length > 0) {
    list.forEach((feat) => {
      connection.queryFeature(feat).then(status => console.log(feat, "is",
 status.enabled)));
    });
  }
}
```

Amazon DCV Web UI を使用する SDK

次のチュートリアルでは、Amazon DCVサーバーに対して認証し、接続して、Amazon DCV Web UI から DCVViewer React コンポーネントをレンダリングする方法を示しますSDK。

トピック

- 前提条件
- ステップ 1: HTMLページを準備する
- ステップ 2: DCVViewer React コンポーネントの認証、接続、レンダリングを行う
- AWS-UI から Cloudscape Design System への更新

前提条件

React、ReactDOM、Cloudscape Design Components React、Cloudscape Design Global Styles、Cloudscape Design Design Tokens をインストールする必要があります。

```
$ npm i react react-dom @cloudscape-design/components @cloudscape-design/global-styles
 @cloudscape-design/design-tokens
```

また、Amazon DCV Web Client SDK のダウンロードも必要になります。これを行う方法については、 step-by-stepAmazon DCV Web Client の開始方法 SDK「」を参照してください。

Amazon DCV Web UI の外部依存関係であるため、dcvモジュールをインポートするためのエイリアスを作成する必要がありますSDK。例えば、webpack を使用してウェブアプリケーションをバンドルする場合、以下のように resolve.alias オプションを使用できます。

```
const path = require('path');

module.exports = {
  //...
  resolve: {
    alias: {
      dcv: path.resolve('path', 'to', 'dcv.js'),
    },
  },
};
```

バンドルにロールアップを使用している場合は、@rollup /plugin-alias をインストールして以下のように使用できます。

```
import alias from '@rollup/plugin-alias';
const path = require('path');

module.exports = {
  //...
  plugins: [
    alias({
      entries: [
        { find: 'dcv', replacement: path.resolve('path', 'to', 'dcv.js') },
      ]
    })
  ]
```

```
};
```

ステップ 1: HTMLページを準備する

ウェブページでは、必要な JavaScript モジュールをロードする必要があり、アプリケーションのエントリコンポーネントidがレンダリングされる有効な HTML 要素が必要です。

例:

```
<!DOCTYPE html>
<html lang="en" style="height: 100%;">
  <head>
    <title>DCV first connection</title>
  </head>
  <body style="height: 100%;">
    <div id="root" style="height: 100%;"></div>
    <script type="module" src="index.js"></script>
  </body>
</html>
```

ステップ 2: **DCVViewer** React コンポーネントの認証、接続、レンダリングを行う

このセクションでは、ユーザー認証プロセスを完了する方法、Amazon DCVサーバーを接続する方法、および DCVViewer React コンポーネントをレンダリングする方法を示します。

まず、index.js ファイルから React、ReactDOM、および最上位の App コンポーネントをインポートします。

```
import React from "react";
import ReactDOM from 'react-dom';
import App from './App';
```

アプリケーションの最上位のコンテナノードをレンダリングします。

```
ReactDOM.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
  document.getElementById("root")
```

```
);
```

App.js ファイルで、Amazon DCV Web Client をESMモジュールSDKとしてインポートし、Amazon DCV Web UI から DCVViewer React コンポーネントSDK、Reactおよび Cloudscape Design Global Stylesパッケージをインポートします。

```
import React from "react";
import dcv from "dcv";
import "@cloudscape-design/global-styles/index.css";
import {DCVViewer} from "./dcv-ui/dcv-ui.js";
```

認証が成功した場合、Amazon DCV Server に対して認証しSDK、Amazon DCV Web UI から DCVViewer React コンポーネントをレンダリングする方法の例を次に示します。

```
const LOG_LEVEL = dcv.LogLevel.INFO;
const SERVER_URL = "https://your-dcv-server-url:port/";
const BASE_URL = "/static/js/dcvjs";

let auth;

function App() {
  const [authenticated, setAuthenticated] = React.useState(false);
  const [sessionId, setSessionId] = React.useState('');
  const [authToken, setAuthToken] = React.useState('');
  const [credentials, setCredentials] = React.useState({});

  const onSuccess = (_, result) => {
    var { sessionId, authToken } = { ...result[0] };

    console.log("Authentication successful.");

    setSessionId(sessionId);
    setAuthToken(authToken);
    setAuthenticated(true);
    setCredentials({});
  }

  const onPromptCredentials = (_, credentialsChallenge) => {
    let requestedCredentials = {};

    credentialsChallenge.requiredCredentials.forEach(challenge =>
 requestedCredentials[challenge.name] = "");
```

```
    setCredentials(requestedCredentials);
  }

  const authenticate = () => {
    dcv.setLogLevel(LOG_LEVEL);

    auth = dcv.authenticate(
      SERVER_URL,
      {
        promptCredentials: onPromptCredentials,
        error: onError,
        success: onSuccess
      }
    );
  }

  const updateCredentials = (e) => {
    const { name, value } = e.target;
    setCredentials({
      ...credentials,
      [name]: value
    });
  }

  const submitCredentials = (e) => {
    auth.sendCredentials(credentials);
    e.preventDefault();
  }

  React.useEffect(() => {
    if (!authenticated) {
      authenticate();
    }
  }, [authenticated]);

  const handleDisconnect = (reason) => {
    console.log("Disconnected: " + reason.message + " (code: " + reason.code + ")");
    auth.retry();
    setAuthenticated(false);
  }

  return (
    authenticated ?
    <DCVViewer
```

```
      dcv={{
        sessionId: sessionId,
        authToken: authToken,
        serverUrl: SERVER_URL,
        baseUrl: BASE_URL,
        onDisconnect: handleDisconnect,
        logLevel: LOG_LEVEL
      }}
      uiConfig={{
        toolbar: {
          visible: true,
          fullscreenButton: true,
          multimonitorButton: true,
        },
      }}
    />
    :
    <div
      style={{
        height: window.innerHeight,
        backgroundColor: "#373737",
        display: 'flex',
        alignItems: 'center',
        justifyContent: 'center',
      }}
    >
      <form>
        <fieldset>
          {Object.keys(credentials).map((cred) => (
            <input
              key={cred}
              name={cred}
              placeholder={cred}
              type={cred === "password" ? "password" : "text"}
              onChange={updateCredentials}
              value={credentials[cred]}
            />
          ))}
        </fieldset>
        <button
          type="submit"
          onClick={submitCredentials}
        >
          Login
```

```
        </button>
      </form>
    </div>
  );
}

const onError = (_, error) => {
  console.log("Error during the authentication: " + error.message);
}

export default App;
```

promptCredentials 関数、error 関数、success 関数は、必須コールバック関数であり、認証プロセスで定義する必要があります。。

Amazon DCVサーバーが認証情報の入力を求められた場合、promptCredentialsコールバック関数は Amazon DCVサーバーから要求された認証情報チャレンジを受け取ります。Amazon DCVサーバーがシステム認証を使用するように設定されている場合、認証情報はユーザー名とパスワードの形式で提供する必要があります。

認証に失敗すると、errorコールバック関数は Amazon DCVサーバーからエラーオブジェクトを受け取ります。

認証が成功すると、successコールバック関数は、ユーザーが Amazon DCVサーバーで接続することを許可されている各セッションのセッション ID (sessionId) と認証トークン (authToken) を含むカップルの配列を受け取ります。上記のコードサンプルは、認証に成功すると React ステータスを更新し、DCVViewer コンポーネントをレンダリングします。

このコンポーネントで受け入れられるプロパティの詳細については、「Amazon DCV Web UI SDKリファレンス」を参照してください。

自己署名証明書の詳細については、「自己署名証明書によるリダイレクトの説明」をご覧ください。

AWS-UI から Cloudscape Design System への更新

SDK バージョン 1.3.0 以降、DCVViewerコンポーネントを AWS-UI から進化した Cloudscape Design に更新しました。

Cloudscape は AWS-UI とは異なるビジュアルテーマを使用しますが、基盤となるコードベースは同じままです。したがって、DCVViewer ベースのアプリケーションの移行は容易です。移行するに

は、インストールした AWS-UI 関連のNPMパッケージを、関連する Cloudscape パッケージに置き換えます。

AWS-UI パッケージ名	Cloudscape パッケージ名
@awsui/components-react	@cloudscape-design/components
@awsui/global-styles	@cloudscape-design/global-styles
@awsui/collection-hooks	@cloudscape-design/collection-hooks
@awsui/design-tokens	@cloudscape-design/design-tokens

移行の詳細については、AWS-UI GitHub ドキュメントページ を参照してください。

SDK リファレンス

このセクションでは、Amazon DCV Web Client の説明、構文、使用例について説明しますSDK。

トピック

- DCV モジュール
- 接続クラス
- 認証クラス
- リソースクラス
- Amazon DCV Web UI SDK

DCV モジュール

DCV プロトコルのクライアント側を実装するモジュール。

エクスポーズ

- 方法
- メンバー
- データ型とコールバックの定義

方法

リスト

- authenticate(url, callbacks) → {Authentication}
- connect(config) → {Promise.<接続 >|約束。<{code: ConnectionErrorCode、メッセージ: string}>}
- setLogHandler (ハンドラー) → {void}
- setLogLevel (レベル) → {void}

authenticate(url, callbacks) → {Authentication}

指定された Amazon DCVサーバーエンドポイントの認証プロセスを開始します。

[パラメータ:]

名前	型	説明
`url`	string	実行中の Amazon DCV サーバーのホスト名とポートを次の形式で指定します。 `https://dcv_host_address:port` 例: `https://my-dcv-server:8443` 。
`callbacks`	authenticationCallbacks	認証プロセス中に呼び出すことができるコールバック。

戻り値:

- Authentication オブジェクト。

タイプ

Authentication

connect(config) → {Promise.<接続 >|約束。<{code: ConnectionErrorCode、メッセージ: string}>}

指定された Amazon DCVサーバーエンドポイントに接続します。接続が成功した場合は Connection オブジェクトを返します。接続が失敗した場合はエラーオブジェクトを返します。

[パラメータ:]

名前	型	説明
`config`	ConnectionConfig	ConnectionConfig オブジェクト。

戻り値:

- Connection オブジェクト、またはエラーオブジェクト。

タイプ

約束。 < 接続 > | Promise.<{code: ConnectionErrorCode、メッセージ: string}>

setLogHandler (ハンドラー) → {void}

カスタムログハンドラ関数を設定します。デフォルトのログハンドラをオーバーライドすると、ブラウザコンソールでデバッグするときに元のログエントリの位置が失われます。

[パラメータ:]

名前	型	説明
`handler`	function	カスタムログハンドラ関数。ハンドラー関数には、レベル (数値) 、 levelName (文字列) 、ドメイン (文字列) 、およびメッセージ (文字列) が含まれます。

戻り値:

タイプ

void

setLogLevel (レベル) → {void}

ログレベルを設定します。これは、デフォルトのログハンドラが使用されている場合にのみ必要です。

[パラメータ:]

名前	型	説明
`level`	LogLevel	使用するログレベル。

戻り値:

タイプ

void

メンバー

リスト

- （定数）AudioError :AudioErrorCode
- （定数）AuthenticationError :AuthenticationErrorCode
- （定数）ChannelError :ChannelErrorCode
- （定数）ClosingReasonError :ClosingReasonErrorCode
- （定数）ConnectionError :ConnectionErrorCode
- （定数）CustomChannelError :CustomChannelErrorCode
- （定数）DisplayConfigError :DisplayConfigErrorCode
- （定数）FileStorageError :FileStorageErrorCode
- （定数）LogLevel :LogLevel
- （定数）MultiMonitorError :MultiMonitorErrorCode
- （定数）ResolutionError :ResolutionErrorCode
- （定数）TimezoneRedirectionError :TimezoneRedirectionErrorCode
- （定数）TimezoneRedirectionSetting :TimezoneRedirectionSettingCode
- （定数）TimezoneRedirectionStatus :TimezoneRedirectionStatusCode
- (定数) バージョン
- （定数）ScreenshotError :ScreenshotErrorCode
- （定数）WebcamError :WebcamErrorCode

（定数）AudioError :AudioErrorCode

AudioError コード列挙型。

タイプ:

- AudioErrorCode

（定数）AuthenticationError :AuthenticationErrorCode

AuthenticationError コード列挙型。

タイプ:

- AuthenticationErrorCode

（定数）ChannelError :ChannelErrorCode

ChannelError コード列挙型。

タイプ:

- ChannelErrorCode

（定数）ClosingReasonError :ClosingReasonErrorCode

ClosingReasonError コード列挙型。

タイプ:

- ClosingReasonErrorCode

（定数）ConnectionError :ConnectionErrorCode

ConnectionError コード列挙型。

タイプ:

- ConnectionErrorCode

（定数）CustomChannelError :CustomChannelErrorCode

CustomChannelError コード列挙型。

タイプ:

- CustomChannelErrorCode

（定数）DisplayConfigError :DisplayConfigErrorCode

DisplayConfigError コード列挙型。

タイプ:

- DisplayConfigErrorCode

（定数）FileStorageError :FileStorageErrorCode

FileStorageError コード列挙型。

タイプ:

- FileStorageErrorCode

（定数）LogLevel :LogLevel

使用可能なSDKログレベル。

タイプ:

- LogLevel

（定数）MultiMonitorError :MultiMonitorErrorCode

MultiMonitorError コード列挙型。

タイプ:

- MultiMonitorErrorCode

（定数） ResolutionError :ResolutionErrorCode

ResolutionError コード列挙型。

タイプ:

- ResolutionErrorCode

（定数） TimezoneRedirectionError :TimezoneRedirectionErrorCode

TimezoneRedirectionError コード列挙型。

タイプ:

- TimezoneRedirectionErrorCode

（定数） TimezoneRedirectionSetting :TimezoneRedirectionSettingCode

TimezoneRedirectionSetting コード列挙型。

タイプ:

- TimezoneRedirectionSettingCode

（定数） TimezoneRedirectionStatus :TimezoneRedirectionStatusCode

TimezoneRedirectionStatus コード列挙型。

タイプ:

- TimezoneRedirectionStatusCode

(定数) バージョン

メジャー、マイナー、パッチ、リビジョン、拡張、 を含む Amazon DCVバージョンversionStr。

プロパティ:

名前	型	説明
major	integer	メジャーバージョン番号。
minor	integer	マイナーバージョン番号。
patch	integer	パッチバージョン番号。
revision	integer	リビジョン番号。
extended	string	拡張文字列。
versionStr	string	メジャー番号、マイナー番号、パッチ番号、リビジョン番号を形式 major.minor.patch+build.revision で連結したもの。

（定数）ScreenshotError :ScreenshotErrorCode

ScreenshotError コード列挙型。

タイプ:

- ScreenshotErrorCode

（定数）WebcamError :WebcamErrorCode

WebcamError コード列挙型。

タイプ:

- WebcamErrorCode

データ型とコールバックの定義

リスト

- AudioErrorCode
- authenticationCallbacks
- AuthenticationErrorCode
- authErrorCallback (認証、エラー)
- authPromptCredentialsコールバック (認証、チャレンジ)
- authSuccessCallback (認証、authenticationData)
- Channel
- ChannelErrorCode
- clipboardEventCallback (イベント)
- ClosingReasonErrorCode
- Colorspace
- connectionCallbacks
- ConnectionConfig
- ConnectionErrorCode
- createDirectory (パス)
- CustomChannelErrorCode
- dataChannelCallback (情報)
- deleteFile (パス)
- deviceChangeEventコールバック ()
- disconnectCallback (理由)
- displayAvailabilityCallback (ステータス、displayId)
- DisplayConfigErrorCode
- displayLayoutCallback (serverWidth、serverHeight、ヘッド)
- 機能
- featuresUpdateCallback(featuresList)
- fileDownloadCallback(fileResource)
- filePrintedCallback(printResource)
- filestorage
- filestorageEnabledCallback (有効)
- FileStorageErrorCode

- firstFrameCallback (resizeEnabled、 relativeMouseMode有効、displayId ）
- idleWarningNotificationコールバック (disconnectionDateTime ）
- collaboratorListCallback (コラボレーター ）
- licenseNotificationCallback (通知 ）
- list(path)
- LogLevel
- モニタリング
- MultiMonitorErrorCode
- qualityIndicatorStateコールバック (状態 ）
- renameDirectory(src、dest)
- renameFile(src、dest)
- ResolutionErrorCode
- retrieveFile (パス ）
- screenshotCallback (スクリーンショット ）
- ScreenshotErrorCode
- serverInfo
- stats
- storeFile (ファイル、dir)
- TimezoneRedirectionErrorCode
- TimezoneRedirectionSettingCode
- TimezoneRedirectionStatusCode
- WebcamErrorCode

AudioErrorCode

DCV モジュールで使用できる AudioError コード列挙

- `SETTING_AUDIO_FAILED`
- `CHANNEL_NOT_AVAILABLE`

タイプ:

- number

authenticationCallbacks

認証コールバック

タイプ:

- オブジェクト

プロパティ:

名前	型	説明
promptCredentials	authPromptCredentialsコールバック	ユーザーが認証情報のチャレンジを受けたときに呼び出されるコールバック関数。
error	authErrorCallback	認証が失敗したときに呼び出されるコールバック関数。
success	authSuccessCallback	認証が成功したときに呼び出されるコールバック関数。

AuthenticationErrorCode

DCV モジュールで使用できる AuthenticationError コード列挙

- INVALID_MESSAGE
- UNKNOWN_AUTH_MODE
- SESSION_NOT_AVAILABLE
- NO_SESSIONS
- WRONG_CREDENTIALS
- SASL_CHALLENGE
- SASL_AUTH_MECHANISM
- FAILED_COMMUNICATION
- AUTHENTICATION_REJECTED
- GENERIC_ERROR
- WRONG_CREDENTIALS_FORMAT

- WRONG_CREDENTIALS_TYPE
- UNREQUESTED_CREDENTIALS
- MISSING_CREDENTIAL

タイプ:

- number

authErrorCallback (認証、エラー)

認証が失敗したときに呼び出されるコールバック関数。

[パラメータ:]

名前	型	説明
authentication	Authentication	Authentication オブジェクト。
error	オブジェクト	認証プロセスによって発生したエラーオブジェクト。 名前 / 型 / 説明 code / AuthenticationErrorCode / エラーコードです。 message / string / エラーメッセージです。

authPromptCredentialsコールバック (認証、チャレンジ)

ユーザーが認証情報のチャレンジを受けたときに呼び出されるコールバック関数。ユーザーは、要求された認証情報を提供してチャレンジに応答する必要があります。

[パラメータ:]

名前	型	説明
authentication	Authentication	Authentication オブジェクト。
challenge	オブジェクト	チャレンジ。 名前: requiredCredentials 型: Array.<Object> 説明: リクエストされた認証情報オブジェクトの配列。 名前: challenge 型: string 説明: リクエストされた認証情報の

名前	型	説明
		名前 型 説明 名前。 ts リクエストされた認証情報の型。

authSuccessCallback (認証、authenticationData)

認証が成功したときに呼び出されるコールバック関数。

[パラメータ:]

名前	型	説明
`authentication`	Authentication	Authentication オブジェクト。

名前	型	説明
authenticationData	Array.<Object>	Amazon DCVセッションIDs トークンと認証トークンを含むオブジェクトの配列。 名前 / 型 / 説明 sessior / string / Amazon DCV セッションID。 authTok / string / Amazon DCV セッションの認証トークン。

Channel

指定できる使用可能なチャンネル。

タイプ:

- "clipboard" | "display" | "input" | "audio" | "filestorage"

ChannelErrorCode

DCV モジュールで使用できる ChannelError コード列挙

- ALREADY_OPEN
- INITIALIZATION_FAILED
- REJECTED

タイプ:

- number

clipboardEventCallback (イベント)

`clipboardEvent` の生成時に呼び出されるコールバック関数。

[パラメータ:]

<table>
<tr><th>名前</th><th>型</th><th>説明</th></tr>
<tr><td>event</td><td>オブジェクト</td><td>クリップボードイベントに関する情報。
<table>
<tr><th>名前</th><th>型</th><th>属性</th><th>説明</th></tr>
<tr><td>name</td><td>確立された | コピー | 貼り付け | dataSi lert | autoC one | newDa ailable | autoPa Done | remote</td><td></td><td>常に存在します。イベントの名前。</td></tr>
</table>
</td></tr>
</table>

名前	型	説明

名前	型	属性	説明
	or | pasteA lableD		
clipb Data	Object | string		クリップボード内のデータ。
autoC	boolea	<optio >	セッションクリップボードからローカルクライアントクリップボードへの自動コピー

名前	型	説明
		名前 型 属性 説明 が有効かどうかを示します。 maxDa ze 数値 <optio > クリップボードに配置できるデータの最大量。 erroı string <optio > エラー情報(該当する場合)。

ClosingReasonErrorCode

DCV モジュールで使用できる ClosingReasonError コード列挙

- TRANSPORT_ERROR
- NO_ERROR
- GENERIC_ERROR
- INTERNAL_SERVER_ERROR
- PROTOCOL_ERROR
- AUTHORIZATION_DENIED
- AUTHORIZATION_REVOKED
- ACCESS_REJECTED
- IDLE_TIMEOUT_EXPIRED
- DISCONNECT_BY_OWNER
- DISCONNECT_BY_USER
- EVICTED
- EXTERNAL_PROTOCOL_CONNECTION_EVICTED
- DISCONNECTION_REQUESTED

タイプ:

- number

Colorspace

指定できる使用可能な色空間。

タイプ:

- RGB「」 |YUV「_REC601」 |YUV「_REC709」

connectionCallbacks

接続エラーが発生した場合に呼び出すことができるコールバック。

タイプ:

- オブジェクト

プロパティ:

名前	型	説明
disconnect	disconnectCallback	接続の終了時に呼び出されるコールバック関数。
displayLayout	displayLayoutCallback	ディスプレイレイアウトまたは解像度が変更されたときに呼び出されるコールバック関数。
displayAvailability	displayAvailabilityCallback	ディスプレイの可用性が変更されたときに呼び出されるコールバック関数。
firstFrame	firstFrameCallback	Amazon DCVサーバーから最初のフレームを受信したときに呼び出されるコールバック関数。
filePrinted	filePrintedCallback	Amazon DCVサーバーにファイルが印刷されたときに呼び出されるコールバック関数。
fileDownload	fileDownloadCallback	Amazon DCVサーバーからファイルをダウンロードする準備ができたときに呼び出されるコールバック関数。
dataChannel	dataChannelCallback	Amazon DCVサーバーがデータチャネルの可用性に関する通知を送信するときに呼び出されるコールバック関数。
licenseNotification	licenseNotificationCallback	Amazon DCVサーバーがライセンス状態に関する通知を送

名前	型	説明
		信するときに呼び出されるコールバック関数。
idleWarningNotification	idleWarningNotificationコールバック	Amazon DCVサーバーがアイドルタイムアウト警告を送信するときに呼び出されるコールバック関数。
collaboratorList	collaboratorListCallback	Amazon DCVサーバーがコラボレーターのリストを送信するときに呼び出されるコールバック関数 (Amazon DCV Web Client SDKバージョン 1.1.0 以降) 。
qualityIndicatorState	qualityIndicatorStateコールバック	接続品質インジケータで状態が変化したときに呼び出されるコールバック関数。
filestorageEnabled	filestorageEnabledCallback	ファイルストレージが有効または無効になったときに呼び出されるコールバック関数。
featuresUpdate	featuresUpdateCallback	機能のステータスが変化したときに呼び出されるコールバック関数。
clipboardEvent	clipboardEventCallback	clipboardEvent の生成時に呼び出されるコールバック関数。
deviceChangeEvent	deviceChangeEventコールバック	deviceChange イベントがトリガーされたときに呼び出されるコールバック関数。

名前	型	説明
screenshot	screenshotCallback	screenshot が使用可能であるときに呼び出されるコールバック関数。

ConnectionConfig

Amazon DCV接続の設定。

タイプ:

- オブジェクト

プロパティ:

名前	型	説明
url	string	実行中の Amazon DCV サーバーのホスト名とポートを次の形式で指定します。 https://dcv_host_address:port 例: https://my-dcv-server:8443 。
sessionId	string	Amazon DCVセッション ID。
authToken	string	セッションへの接続に使用する認証トークン。
baseUrl	string	SDK ファイルをロードURLする絶対または相対。
resourceBaseUrl	string	DCV リソースURLにアクセスする絶対または相対。

名前	型	説明
enabledChannels	Array.<Channel>	有効化できるチャンネルのリストを示します。指定しない場合、または空の配列を指定した場合、デフォルトによりすべての使用可能なチャンネルになります。
losslessColorspace	Colorspace	使用される色空間を示します。指定しない場合、デフォルトはRGB「」になります。
divId	string	HTML DOM がリモートストリームでキャンバスSDKを作成する 内のdivオブジェクトの ID。
volumeLevel	integer	希望するボリュームレベル。有効範囲は 0 ~ 100 です。
clipboardAutoSync	ブール値	Amazon DCVセッションクリップボードからローカルクライアントクリップボードへの自動コピーが、互換性のあるウェブブラウザで有効になっているかどうかを示します。
dynamicAudioTuning	ブール値	接続が確立されたときに、Amazon DCVサーバーのオーディオ設定に基づいてオーディオを動的に調整するかどうかを示します。
clientHiDpiScaling	ブール値	クライアントの に基づいてキャンバスをスケーリングするかどうかを示しますDPI。

名前	型	説明
highColorAccuracy	ブール値	高い色精度が使用可能な場合に使用するかどうかを示します。指定されない場合、デフォルトは false です。
enableWebCodecs	ブール値	利用可能な場合 WebCodecs 、 を使用するかどうかを示します。指定されない場合のデフォルト値は false です。
observers	connectionCallbacks	接続に関連するイベントを呼び出すためのコールバック関数。
callbacks	connectionCallbacks	observers プロパティと同じですが、各コールバックに Connection オブジェクトが最初のパラメータとして含まれます。

ConnectionErrorCode

DCV モジュールで使用できる ConnectionError コード列挙

- ALREADY_OPEN
- INVALID_CONFIG
- INITIALIZATION_FAILED
- REJECTED
- MAIN_CHANNEL_ALREADY_OPEN
- GENERIC_ERROR (DCVサーバー 2021.0 以降)
- INTERNAL_SERVER_ERROR (DCVサーバー 2021.0 以降)
- AUTHENTICATION_FAILED (DCVサーバー 2021.0 以降)
- PROTOCOL_ERROR (DCVサーバー 2021.0 以降)
- INVALID_SESSION_ID (DCVサーバー 2021.0 以降)

- INVALID_CONNECTION_ID (DCVサーバー 2021.0 以降)
- CONNECTION_LIMIT_REACHED (DCVサーバー 2021.0 以降)
- SERVER_UNREACHABLE (DCVサーバー 2022.1 以降)
- GATEWAY_BUSY
- UNSUPPORTED_CREDENTIAL (DCVサーバー 2022.2 以降)
- TRANSPORT_ERROR

タイプ:

- number

createDirectory (パス)

[パラメータ:]

名前	型	説明
path	string	ディレクトリを作成するサーバーの絶対パス。ターゲットディレクトリの名前も含まれます。

CustomChannelErrorCode

DCV モジュールで使用できる CustomChannelError コード列挙

- TRANSPORT_ERROR

タイプ:

- number

dataChannelCallback (情報)

Amazon DCVサーバーがデータチャネルの可用性に関する通知を送信するときに呼び出されるコールバック関数。

[パラメータ:]

名前	型	説明
info	オブジェクト	データチャンネルに関する情報。 名前 / 型 / 説明 name / string / データチャンネルの名前。 token / string / データチャンネルの認証トークン。

deleteFile (パス)

[パラメータ:]

名前	型	説明
path	string	削除するファイルを識別するサーバーの絶対パス。

deviceChangeEventコールバック ()

deviceChange イベントがトリガーされたときに呼び出されるコールバック関数。

disconnectCallback (理由)

接続の終了時に呼び出されるコールバック関数。

[パラメータ:]

名前	型	説明
reason	オブジェクト	切断の理由。 名前 / 型 / 説明 code / 数値 / 理由コード。 message / string / 理由メッセージ。

displayAvailabilityCallback (ステータス、displayId)

ディスプレイの可用性が変更されたときに呼び出されるコールバック関数。

[パラメータ:]

名前	型	説明
status	オブジェクト	ディスプレイのステータス。 名前 / 型 / 説明 enabled / ブール型 / ディスプレイが有効かどうかを示します。

名前	型	説明
		名前 / 型 / 説明 closed / ブール値 / ディスプレイが閉じているかどうかを示します。
displayId	数値	ディスプレイの識別子。

DisplayConfigErrorCode

DCV モジュールで使用できる DisplayConfigError コード列挙

- INVALID_ARGUMENT
- UNSUPPORTED_OPERATION
- NO_CHANNEL

タイプ:

- number

displayLayoutCallback (serverWidth、serverHeight、ヘッド)

ディスプレイレイアウトまたは解像度が変更されたときに呼び出されるコールバック関数。

[パラメータ:]

名前	型	説明
serverWidth	数値	プライマリディスプレイの幅 (ピクセル)。

名前	型	説明
serverHeight	数値	プライマリディスプレイの高さ (ピクセル)。
heads	Array.<Monitor>	Amazon DCVサーバーでサポートされているディスプレイヘッド。

機能

機能値。

- display - シングルディスプレイビデオストリームの可用性を示します。
- display-multi - マルチディスプレイビデオストリームの可用性を示します。
- high-color-accuracy - 高い色精度の可用性を示します (Amazon DCV Web Client SDKバージョン 1.1.0 以降) 。
- mouse - マウス機能の可用性を示します。
- keyboard - キーボード機能の可用性を示します。
- keyboard-sas - SASシーケンス (コントロール + Alt + 削除) 機能の可用性を示します。
- relative-mouse - 相対マウスモードの可用性を示します。
- clipboard-copy - Amazon DCVサーバーからクライアントへのクリップボードコピー機能の可用性を示します。
- clipboard-paste - クライアントから Amazon DCVサーバーへのクリップボード貼り付け機能の可用性を示します。
- audio-in - マイクを使用したオーディオ入力機能の可用性を示します。
- audio-out - オーディオ再生機能の可用性を示します。
- webcam - ウェブカメラストリーミング機能の可用性を示します。
- file-download - Amazon DCVサーバーからクライアントへのファイルダウンロード機能の可用性を示します。
- file-upload - クライアントから Amazon DCVサーバーへのファイルアップロード機能の可用性を示します。
- timezone-redirection - タイムゾーンリダイレクト機能の可用性を示します (Amazon DCV Web Client SDKバージョン 1.3.0 以降) 。

タイプ:

- string

featuresUpdateCallback(featuresList)

機能のステータスが変化したときに呼び出されるコールバック関数。

[パラメータ:]

名前	型	説明
`featuresList`	Array.<feature>	変化した一連の機能。

fileDownloadCallback(fileResource)

Amazon DCVサーバーからファイルをダウンロードする準備ができたときに呼び出されるコールバック関数。

[パラメータ:]

名前	型	説明
`fileResource`	オブジェクト	ダウンロード可能になったファイルに関する情報。 名前 / 型 / 説明 `id` / string / ファイルの識別子。 `url` / string / ファイルのダウンロードURLに

名前	型	説明
		名前 / 型 / 説明 使用する 。 domain / string / リソースドメイン。 token / string / ファイルのダウンロードに使用する認証トークン。トークンは にも含まれています URL。

filePrintedCallback(printResource)

Amazon DCVサーバーにファイルが印刷されたときに呼び出されるコールバック関数。

[パラメータ:]

名前	型	説明
printResource	オブジェクト	印刷されたファイルに関する情報。

名前	型	説明
		名前 / 型 / 説明 id / string / 印刷されたファイルの識別子。 url / string / 印刷ファイルのダウンロードURLに使用する 。 domain / string / リソースドメイン。この場合は printer。 token / string / 印刷されたファイルのダウンロードに使用する認証トークン。トーク

名前	型	説明
		名前 型 説明 ンは にも含まれています URL。

filestorage

ファイルシステムにおけるアクションの調査と実行を可能にするオブジェクト。

タイプ:

- オブジェクト

プロパティ:

名前	型	説明
`list`	list	サーバー上の提供されたパスに存在するアイテム (ファイルとディレクトリ) の一覧表示を許可する関数。
`createDirectory`	createDirectory	サーバー上の指定されたパスでのディレクトリ作成を許可する関数。
`retrieveFile`	retrieveFile	サーバー上の指定されたパスでファイルのローカルダウンロードを許可する関数。
`deleteFile`	deleteFile	サーバー上の指定されたパスでのファイル削除を許可する関数。

名前	型	説明
`renameFile`	renameFile	指定ソースパスから指定送信先パスへのファイル名変更を許可する関数。
`renameDirectory`	renameDirectory	指定ソースパスから絶対送信先パスへのディレクトリ名変更を許可する関数。
`storeFile`	storeFile	サーバー上の指定されたパスへのローカルファイルのアップロードを許可する関数。

filestorageEnabledCallback (有効)

ファイルストレージが有効になったときに呼び出されるコールバック関数。Internet Explorer 11 のレイジーチャンネルのみ。

[パラメータ:]

名前	型	説明
`enabled`	ブール型	ファイルストレージが有効かどうかを示します。

FileStorageErrorCode

DCV モジュールで使用できる FileStorageError コード列挙

- `CANCELLED`
- `ABORTED`
- `INVALID_ARGUMENT`
- `NOT_IMPLEMENTED`
- `ERROR`
- `ALREADY_EXIST`

- NOT_FOUND

タイプ:

- number

firstFrameCallback (resizeEnabled、 relativeMouseMode有効、displayId)

Amazon DCVサーバーから最初のフレームを受信したときに呼び出されるコールバック関数。ディスプレイごとに放出されます。

[パラメータ:]

名前	型	説明
resizeEnabled	ブール型	クライアントディスプレイのレイアウトのサイズ変更がサーバーでサポートされているかどうかを示します。
relativeMouseModeEnabled	ブール値	相対マウスモードがサーバーでサポートされているかどうかを示します。
displayId	数値	ディスプレイの識別子。

idleWarningNotificationコールバック (disconnectionDateTime)

Amazon DCVサーバーがアイドルタイムアウト警告を送信するときに呼び出されるコールバック関数。

[パラメータ:]

名前	型	説明
disconnectionDateTime	日付	切断の日時。

collaboratorListCallback（コラボレーター）

Amazon DCVサーバーがコラボレーターのリストを送信するときに呼び出されるコールバック関数。

[パラメータ:]

名前	型	説明
collaborators	Array.<Object>	コラボレーターに関する情報を含むオブジェクトのリスト。

名前	型	説明
usernam	string	コラボレーターのユーザー名。
owner	ブール値	コラボレーターがセッションオーナーであるかどうかを示します。
connectnId	数値	サーバーによって接続に

名前	型	説明
		名前 / 型 / 説明 割り当てられた ID を示します。

licenseNotificationCallback（通知）

Amazon DCVサーバーがライセンス状態に関する通知を送信するときに呼び出されるコールバック関数。

[パラメータ:]

名前	型	説明
`notification`	オブジェクト	通知。 名前 / 型 / 説明 `product` string DCV 製品。 `status` string ライセンスのステータス。 `message` string メッセージ。 `leftDay` 数値 ライセンスの有効期

名前	型	説明
		名前 / 型 / 説明 限が切れるまでの日数。 /
		isDemo / ブール値 / ライセンスがデモライセンスかどうかを示します。
		numUnlicensed / 数値 / ライセンスを取得していない接続の数。
		licensingMode / string / ライセンス取得モード。
		documentationUrl / string / ドキュメントURLの 。

list(path)

[パラメータ:]

名前	型	説明
path	string	コンテンツを一覧表示するサーバー上の絶対パス。

LogLevel

使用可能なSDKログレベル。

タイプ:

- TRACE | DEBUG | INFO | WARN | ERROR | SILENT

モニタリング

タイプ:

- オブジェクト

プロパティ:

名前	型	説明
name	string	ディスプレイヘッドの名前。
rect	オブジェクト	ディスプレイヘッドに関する情報。 名前 / 型 / 説明 x / 数値 / ディスプレイヘッド

<table>
<tr><th>名前</th><th>型</th><th>説明</th></tr>
<tr><td></td><td></td><td>
<table>
<tr><th>名前</th><th>型</th><th>説明</th></tr>
<tr><td></td><td></td><td>の最初の x 座標。</td></tr>
<tr><td>y</td><td>数値</td><td>ディスプレイヘッドの最初の y 座標。</td></tr>
<tr><td>width</td><td>数値</td><td>ディスプレイヘッドの幅 (ピクセル)。</td></tr>
<tr><td>height</td><td>数値</td><td>ディスプレイヘッドの高さ (ピクセル)。</td></tr>
</table>
</td></tr>
<tr><td>primary</td><td>ブール値</td><td>ディスプレイヘッドがプライマリディスプレイヘッドかどうかを示します。これは、リモートオペレーティングシステムがある場合はそこから決定されます。</td></tr>
</table>

名前	型	説明
dpi	数値	ディスプレイヘッドDPIの 。

MultiMonitorErrorCode

DCV モジュールで使用できる MultiMonitorError コード列挙

- NO_DISPLAY_CHANNEL
- MAX_DISPLAY_NUMBER_REACHED
- INVALID_ARGUMENT
- DISPLAY_NOT_OPENED_BY_SERVER
- REQUEST_TIMEOUT
- GENERIC_ERROR
- NO_ERROR

タイプ:

- number

qualityIndicatorStateコールバック (状態)

接続品質インジケータで状態が変化したときに呼び出されるコールバック関数。

[パラメータ:]

名前	型	説明
state	Array.<Object>	接続品質に関する情報。 名前 / 型 / 説明 name / string / インジケータの名前。

名前	型	説明
		名前 / 型 / 説明 status / NORMAL | WARNIN | CRITICA / ステータスの説明。 changed / ブール値 / ステータスが変化したかどうかを示します。

renameDirectory(src、dest)

[パラメータ:]

名前	型	説明
src	string	名前を変更するディレクトリを識別するサーバー上の絶対ソースパス。
dest	string	ターゲットパスとディレクトリ名を指定するサーバー上の絶対送信先パス。

renameFile(src、dest)

[パラメータ:]

名前	型	説明
`src`	string	名前を変更するファイルを識別するサーバー上の絶対ソースパス。
`dest`	string	ターゲットパスとファイル名を指定するサーバー上の絶対送信先パス。

ResolutionErrorCode

DCV モジュールで使用できる ResolutionError コード列挙

- `INVALID_ARGUMENT`
- `NO_CHANNEL`
- `NOT_IMPLEMENTED`

タイプ:

- number

retrieveFile (パス)

[パラメータ:]

名前	型	説明
`path`	string	ローカルでダウンロードするファイルを識別するサーバー上の絶対パス。

screenshotCallback (スクリーンショット)

screenshotが使用可能であるときに呼び出されるコールバック関数。

[パラメータ:]

名前	型	説明
`screenshot`	byte[]	形式のスクリーンショットバッファPNG、またはスクリーンショットの取得に失敗したnull場合。

ScreenshotErrorCode

DCV モジュールで使用できる ScreenshotError コード列挙

- `NO_CHANNEL`
- `GENERIC_ERROR`

タイプ:

- number

serverInfo

タイプ:

- オブジェクト

プロパティ:

名前	型	説明
`name`	string	ソフトウェアの名前。
`version`	オブジェクト	ソフトウェアのバージョン番号。

<table>
<tr><th>名前</th><th>型</th><th>説明</th></tr>
<tr><td></td><td></td><td>
<table>
<tr><th>名前</th><th>型</th><th>説明</th></tr>
<tr><td>major</td><td>数値</td><td>メジャーバージョン番号。</td></tr>
<tr><td>minor</td><td>数値</td><td>マイナーバージョン番号。</td></tr>
<tr><td>revisic</td><td>数値</td><td>リビジョンバージョン番号。</td></tr>
</table>
</td></tr>
<tr><td>os</td><td>string</td><td>OS。</td></tr>
<tr><td>arch</td><td>string</td><td>アーキテクチャ。</td></tr>
<tr><td>hostname</td><td>string</td><td>ホスト名。</td></tr>
</table>

stats

タイプ:

- オブジェクト

プロパティ:

名前	型	説明
fps	数値	現在のフレーム/秒。
traffic	数値	現在のトラフィック (ビット/秒)。
peakTraffic	数値	接続確立以降のトラフィックのピーク (ビット/秒)。
latency	数値	現在のレイテンシー (ミリ秒)。
currentChannels	数値	接続確立後から開いているチャンネルの数。
openedChannels	数値	現在開いているチャンネルの数。
channelErrors	数値	エラーを報告したチャンネルの数。

storeFile (ファイル、dir)

[パラメータ:]

名前	型	説明
file	File	サーバーにアップロードするファイルオブジェクト (詳細については、https://developer.mozilla.org/en-US/docs/Web/API/File を参照してください)。

名前	型	説明
dir	string	ファイルをアップロードするサーバーの絶対パス。

TimezoneRedirectionErrorCode

DCV モジュールで使用できる TimezoneRedirectionError コード列挙

- INVALID_ARGUMENT
- NO_CHANNEL
- USER_CANNOT_CHANGE

タイプ:

- number

TimezoneRedirectionSettingCode

DCV モジュールで使用できる TimezoneRedirectionSetting コード列挙

- ALWAYS_OFF
- ALWAYS_ON
- CLIENT_DECIDES

タイプ:

- number

TimezoneRedirectionStatusCode

DCV モジュールで使用できる TimezoneRedirectionStatus コード列挙

- SUCCESS
- PERMISSION_ERROR
- GENERIC_ERROR

タイプ:

- number

WebcamErrorCode

DCV モジュールで使用できる WebcamError コード列挙

- `SETTING_WEBCAM_FAILED`
- `CHANNEL_NOT_AVAILABLE`

タイプ:

- number

接続クラス

dcv モジュールの `connect` メソッドを呼び出すと得られる接続クラス。使用方法を示した例については「開始方法」セクションを参照してください。

エクスポーズ

- 方法

方法

リスト

- attachDisplay(win, displayConf) → {Promise.<number>|Promise.<{code: MultiMonitorErrorCode、メッセージ: string}>}
- captureClipboardEvents (有効、勝つ、displayId) → {void}
- detachDisplay (displayId) → {void}
- disconnect() → {void}
- disconnectCollaborator (connectionId) → {void}
- enableDisplayQuality更新 (有効) → {void}
- enableHighPixel密度 (有効) → {void}

- enableTimezoneRedirection (有効化) → {Promise|Promise.<{code: TimezoneRedirectionErrorCode、メッセージ: string}>}
- enterRelativeMouseMode() → {void}
- getConnectedDevices() → {約束 < 配列。 < MediaDeviceInfo>>|Promise.<{message: string}>}
- getFileExplorer() → {Promise.<filestorage >|Promise.<{code: ChannelErrorCode、メッセージ: string}>}
- getServerInfo() → {serverInfo}
- getScreenshot() → {Promise|Promise.<{code: ScreenshotErrorCode、メッセージ: string}>}
- getStats() → {stats }
- latchModifierKey (キー、ロケーション、isDown) → {ブール値}
- openChannel (名前、authToken、コールバック、名前空間) → {Promise|Promise.<{code: ChannelErrorCode、メッセージ: string}>}
- queryFeature (featureName) → {Promise.<{enabled: boolean, remote?: string, autoCopy?: boolean, autoPaste?: boolean, serviceStatus?: string, available?: boolean}>|Promise.<{message: string}>}
- registerKeyboardShortcuts (ショートカット) → {void}
- requestDisplayConfig (highColorAccuracy) → {Promise|Promise.<{code: DisplayConfigErrorCode、メッセージ: string}>}
- requestDisplayLayout (レイアウト) → {Promise|Promise.<{code: ResolutionErrorCode、メッセージ: string}>}
- requestResolution (幅、高さ) → {約束|約束。<{コード: ResolutionErrorCode、メッセージ: 文字列}>}
- sendKeyboardEvent (イベント) → {ブール値}
- sendKeyboardShortcut (ショートカット) → {void}
- setDisplayQuality (最小、最大) → {void}
- setDisplayScale (scaleRatio、displayId) → {約束|約束。<{コード: ResolutionErrorCode、メッセージ: 文字列}>} (DEPRECATED)
- setKeyboardQuirks(quirks) → {void}
- setMaxDisplayResolution(maxWidth, maxHeight) → {void}
- setMicrophone (有効化) → {Promise|Promise.<{code: AudioErrorCode、メッセージ: string}>}
- setMinDisplayResolution(minWidth, minHeight) → {void}

- setUploadBandwidth (値) → {number}
- setVolume (ボリューム) → {void}
- setMicrophone (有効化, deviceId) → {約束|約束.<{code: AudioErrorCode, message: string}>}
- setWebcam (有効化, deviceId) → {約束|約束.<{code: WebcamErrorCode, message: string}>}
- syncClipboards() → {ブール値}

attachDisplay(win, displayConf) → {Promise.<number>|Promise.<{code: MultiMonitorErrorCode、メッセージ: string}>}

特定のディスプレイをウィンドウにアタッチします。メインディスプレイはアタッチできません。成功すると、関数が `displayId` を返します。

[パラメータ:]

名前	型	説明
`win`	オブジェクト	ディスプレイをアタッチする必要があるウィンドウ。
`displayConf`	オブジェクト	ディスプレイの設定。 名前 / 型 / 属性 / 説明 displ / 数値 / <optio> / ディスプレイの ID。 displ vName / / / ディスプレイ div の名前。

戻り値:

Promise。拒否された場合、プロミスがエラーオブジェクトを返します。

タイプ

Promise.<number> | Promise.<{code: MultiMonitorErrorCode、メッセージ: string}>

captureClipboardEvents (有効、勝つ、displayId) → {void}

コピー/貼り付けイベントのリッスンを開始または停止します。インタラクティブなクリップボード (貼り付けの場合は常時) の場合、コピー/貼り付けイベントのリッスンを開始する必要があります。例えばモーダルが表示されている場合など、必要なときにのみ、リッスンを開始および停止すると便利です。

[パラメータ:]

名前	型	属性	説明
enabled	ブール型		イベントのリッスンを開始するには、true を指定します。イベントのリッスンを停止するには、false を指定します。
win	オブジェクト	<optional>	イベントをリッスンするウィンドウ。省略した場合、デフォルトのウィンドウが使用されます。
displayId	数値	<optional>	イベントをリッスンするディスプレイの ID。省略した場合、デフォルトのウィンドウディスプレイが使用されます。

戻り値:

タイプ

void

detachDisplay (displayId) → {void}

特定のディスプレイをデタッチします。メインディスプレイはデタッチできません。

[パラメータ:]

名前	型	説明
`displayId`	数値	デタッチするディスプレイのID。

戻り値:

タイプ

void

disconnect() → {void}

Amazon DCVサーバーから切断し、接続を閉じます。

戻り値:

タイプ

void

disconnectCollaborator (connectionId) → {void}

提供された接続 ID (Amazon DCV Web Client SDKバージョン 1.1.0 以降) に接続された共同作業者の切断をリクエストします。

[パラメータ:]

名前	型	説明
connectionId	ブール型	切断される接続の ID。

戻り値:

タイプ

void

enableDisplayQuality更新 (有効) → {void}

更新を受けないストリーミングエリアの表示品質の更新を有効または無効にします。表示品質の更新を無効にすると、帯域幅の使用量は減少しますが、表示品質も低下します。

[パラメータ:]

名前	型	説明
enable	ブール型	表示品質の更新を有効にするには、true を指定します。表示品質の更新を無効にするには、false を指定します。

戻り値:

タイプ

void

enableHighPixel密度 (有効) → {void}

クライアントで高ピクセル密度を有効または無効にします。

[パラメータ:]

名前	型	説明
enable	ブール型	高ピクセル密度を有効にするかどうか。

戻り値:

タイプ

void

enableTimezoneRedirection (有効化) → {Promise|Promise.<{code: TimezoneRedirectionErrorCode、メッセージ: string}>}

タイムゾーンリダイレクトを有効または無効にします。有効にすると、クライアントはサーバーのデスクトップタイムゾーンをクライアントのタイムゾーンと一致させるようにサーバーに要求します。

[パラメータ:]

名前	型	説明
enable	ブール型	タイムゾーンリダイレクトを有効にするには、true を指定します。タイムゾーンリダイレクトを無効にするには、false を指定します。

戻り値:

Promise。拒否された場合、プロミスがエラーオブジェクトを返します。

タイプ

Promise.<number> | Promise.<{code: TimezoneRedirectionErrorCode、メッセージ: string}>

enterRelativeMouseMode() → {void}

相対マウスモードを有効にします。

戻り値:

タイプ

void

getConnectedDevices() → {約束 < 配列。 < MediaDeviceInfo>>|Promise.<{message: string}>}

クライアントコンピュータに接続されているメディアデバイスのリストをリクエストします。

戻り値:

成功すると、 MediaDeviceInfo オブジェクトの配列に解決される Promise が返されます。詳細については、https://developer.mozilla.org/en-US/docs/Web/API/ を参照してくださいMediaDeviceInfo。拒否された場合、プロミスがエラーオブジェクトを返します。

タイプ

約束 < 配列。 < MediaDeviceInfo>> | Promise.<{メッセージ: string}>

getFileExplorer() → {Promise.<filestorage >|Promise.<{code: ChannelErrorCode、メッセージ: string}>}

Amazon DCVサーバーのファイルストレージを管理するオブジェクトを取得します。

戻り値:

Promise。満たされた場合はファイルエクスプローラーオブジェクト、拒否された場合はエラーオブジェクトに対して、解決するプロミスを返します。

タイプ

Promise.<filestorage > | Promise.<{code: ChannelErrorCode、メッセージ: string}>

getServerInfo() → {serverInfo}

Amazon DCVサーバーに関する情報を取得します。

戻り値:

サーバーソフトウェアに関する情報。

タイプ

serverInfo

getScreenshot() → {Promise|Promise.<{code: ScreenshotErrorCode、メッセージ: string}>}

リモートデスクトップのスクリーンショットを PNG 形式で取得します。スクリーンショットはscreenshotCallbackオブザーバーに返されます。 は、障害が発生した場合に代わりに返nullされます。

戻り値:

リクエストが処理されたら解決する Promise。拒否された場合、エラーオブジェクトが送られます。

タイプ

Promise | Promise.<{code: ScreenshotErrorCode、メッセージ: string}>

getStats() → {stats }

Amazon DCVサーバーに関する統計を取得します。

戻り値:

ストリーミング統計に関する情報。

タイプ

stats

latchModifierKey (キー、ロケーション、isDown) → {ブール値}

許可された修飾子に対する単一のキーボード `keydown` または `keyup` を送信します。

[パラメータ:]

名前	型	説明
`key`	コントロール | Alt | AltGraph | メタ | OS | シフト	送信するキー。
`location`	KeyboardEvent.location	キーの場所。詳細については、https://developer.mozilla.org/en-US/docs/Web/API/KeyboardEvent/location を参照してください。
`isDown`	ブール値	挿入するキーイベントがキーダウン (`true`) またはキーアップ (`false`) である場合。

戻り値:

リクエストされた組み合わせが有効な場合、関数が `true` を返し、それ以外の場合は `false` を返します。

タイプ

ブール値

openChannel (名前、authToken、コールバック、名前空間) → {Promise| Promise.<{code: ChannelErrorCode、メッセージ: string}>}

Amazon DCV Server で作成された場合は、接続でカスタムデータチャネルを開きます。

[パラメータ:]

名前	型	説明
`name`	string	チャネルの名前。

名前	型	説明
authToken	string	チャンネルへの接続に使用する認証トークン。
callbacks	オブジェクト	onMessage および onClose コールバック関数を呼び出します。
namespace	string	データチャネルの名前空間。Amazon DCV Web Client SDK 1.2.0 および Amazon DCV Server 2022.1 以降で利用可能。

戻り値:

Promise。拒否された場合、エラーオブジェクトが送られます。

タイプ

Promise | Promise.<{code: ChannelErrorCode、メッセージ: string}>

queryFeature (featureName) → {Promise.<{enabled: boolean, remote?: string, autoCopy?: boolean, autoPaste?: boolean, serviceStatus?: string, available?: boolean}>|Promise.<{message: string}>}

特定の Amazon DCVサーバー機能のステータスをクエリします。

[パラメータ:]

名前	型	説明
featureName	機能	クエリを行う機能の名前。

戻り値:

Promise。解決すると、関数は常に enabled プロパティを含む status オブジェクトを返し、場合によっては他のプロパティも返します。拒否された場合、関数が error オブジェクトを返します。

タイプ

{Promise.<{enabled: boolean, remote?: string, autoCopy?: boolean, autoPaste?: boolean, serviceStatus?: string, available?: boolean}> | Promise.<{message: string}>

registerKeyboardShortcuts (ショートカット) → {void}

キーボードショートカットを登録します。

[パラメータ:]

名前	型	説明
shortcuts	Array.<Object>	登録するキーとマッピングの配列。 名前 / 型 / 説明 sequenc / Array.<Object> / 登録するキーボードショートカット。 名前 / 型 / 説明 k / K v / コールバック関数。KeyboardEvent.key が押し

名前	型	説明
		名前 型 説明
		名前 型 説明
		名前 型 説明
		説明したキーの値。詳細については、https://developer.mozilla.org/en-US/docs/Web/API/KeyboardEvent/key を参照し

名前	型	説明
		名前 型 説明 名前 型 説明してください。 location 送信するキーの配列。キーボードにおけるキーの場所。詳細につ

名前	型	説明
		名前 / 型 / 説明 — 名前 / 型 / 説明: いては、https://developer.mozilla.org/en-US/docs/Web/API/KeyboardEvent/location を参照してください。

名前	型	説明

名前	型	説明
output	Array.<Object>	ショートカットによって実行される意図されたアクション。

名前	型	説明
key	KeyboardEvent.key	ユーザーが押したキーの値。詳細については、https://develop

名前	型	説明
		名前 型 説明
		er.mozilla.org/en-US/docs/Web/API/KeyboardEvent/key を参照してください。
		l K v ti 送信するキーの配列 oardE loca

名前	型	説明
		名前 型 説明
		名前 型 説明
		名前 型 説明
		。キーボードにおけるキーの場所。詳細については、https://developer.mozilla.org/en-US/docs/Web/API/

名前	型	説明
		名前 型 説明 Keyboard Event/ location を参照してください。

戻り値:

タイプ

void

requestDisplayConfig (highColorAccuracy) → {Promise|Promise.<{code: DisplayConfigErrorCode、メッセージ: string}>}

Amazon DCV Server から更新された表示設定をリクエストします。Amazon DCV Web Client SDK 1.1.0 および Amazon DCV Server 2022.0 以降で利用可能。

[パラメータ:]

名前	型	説明
highColorAccuracy	ブール型	高い色精度を要求すべきかどうか。

戻り値:

Promise。拒否された場合、プロミスがエラーオブジェクトを返します。

タイプ

Promise | Promise.<{code: DisplayConfigErrorCode、メッセージ: string}>

requestDisplayLayout (レイアウト) → {Promise|Promise.<{code: ResolutionErrorCode、メッセージ: string}>}

接続の更新済みディスプレイレイアウトをリクエストします。

[パラメータ:]

名前	型	説明
layout	Array.<Monitor>	レイアウト内のリクエストされたディスプレイ。

戻り値:

Promise。拒否された場合、エラーオブジェクトが送られます。

タイプ

Promise | Promise.<{code: ResolutionErrorCode、メッセージ: string}>

requestResolution (幅、高さ) → {約束|約束。<{コード: ResolutionErrorCode、メッセージ: 文字列}>}

Amazon DCVサーバーから更新された表示解像度をリクエストします。

[パラメータ:]

名前	型	説明
`width`	数値	リクエストする幅 (ピクセル)。許容される最小値は `0` です。
`height`	数値	リクエストする高さ (ピクセル)。許容される最小値は `0` です。

戻り値:

Promise。拒否された場合、プロミスがエラーオブジェクトを返します。

タイプ

Promise | Promise.<{code: ResolutionErrorCode、メッセージ: string}>

sendKeyboardEvent (イベント) → {ブール値}

キーボードショートカットイベントを送信します。キーボードイベントの詳細については、https://developer.mozilla.org/en-US/docs/Web/API/ KeyboardEventを参照してください。有効なキーボードイベントには、`keydown`、`keypress`、`keyup` が含まれます。これらのイベントの詳細については、https://developer.mozilla.org/en-US/docs/Web/API/KeyboardEvent#events を参照してください。

[パラメータ:]

名前	型	説明
event	KeyboardEvent	送信するキーボードイベント。

戻り値:

イベントが有効でない場合、関数が false を返します。イベントが有効である場合、関数が true を返します。

タイプ

ブール値

sendKeyboardShortcut (ショートカット) → {void}

キーボードショートカットを送信します。この関数を使用して、完全な keydown シーケンスまたは keyup シーケンスを送信します。例えば、Ctrl+Alt + Del を送信すると、全てのキーに対して keydown イベントが送信され、続いて keyup イベントが送信されます。単一のキーを送信したい場合であってもこの関数を使用してください。

[パラメータ:]

名前	型	説明
shortcut	Array.<Object>	送信するキーの配列。 名前 / 型 / 説明 key / Keyboard vent.key / ユーザーが押したキーの値。詳細に

名前	型	説明
		名前 型 説明 については、https://developer.mozilla.org/en-US/docs/Web/API/KeyboardEvent/keyを参照してください。 locatic Keyboarc vent.loca tion 送信するキーの配列。キーボードにおけるキーの場所。詳細については、https://developer.mozilla.org/

名前	型	説明
		名前 型 説明 en-US/docs/Web/API/KeyboardEvent/location を参照してください。

戻り値:

タイプ

void

setDisplayQuality (最小、最大) → {void}

接続に使用する画質を設定します。有効範囲は `0` ~ `100` で、`1` が最低画質、`100` が最高画質になります。現在の値を維持するには `0` を指定します。

[パラメータ:]

名前	型	属性	説明
`min`	数値		最低画質。
`max`	数値	<optional>	最高画質。

戻り値:

タイプ

void

setDisplayScale (scaleRatio、displayId) → {約束|約束。<{コード: ResolutionErrorCode、メッセージ: 文字列}>} (DEPRECATED)

バージョン 1.3.0 以降は非推奨です。もうディスプレイのスケールを設定する必要はなくなりました。マウス座標は内部で自動的に管理されます。

ディスプレイがクライアント側でスケーリングされていることを Amazon に通知DCVします。これを使用して、クライアントの表示比率に合わせてマウスイベントのスケーリングを行う必要があることをサーバーに通知します。

[パラメータ:]

名前	型	説明
`scaleRatio`	フロート	使用するスケーリング比率。厳密に正の数である必要があります。
`displayId`	数値	スケーリングを行うディスプレイの ID。

戻り値:

Promise。拒否された場合、プロミスがエラーオブジェクトを返します。

タイプ

Promise | Promise.<{code: ResolutionErrorCode、メッセージ: string}>

setKeyboardQuirks(quirks) → {void}

クライアントコンピュータのキーボード特異性を設定します。

[パラメータ:]

名前	型	説明
quirks	オブジェクト	有効または無効にするキーボード特異性。 名前 / 型 / 説明 macOptiToAlt / ブール型 / macOS の Option キーを Alt にマップするには、true を指定します。それ以外の場合は、false を指定します。 macCommdToCtrl / ブール値 / macOS の Command キーを Ctrl にマップするには、true を指

名前	型	説明
		名前 / 型 / 説明 定します。それ以外の場合は、`false`を指定します。

戻り値:

タイプ

void

setMaxDisplayResolution(maxWidth, maxHeight) → {void}

接続に使用する最高表示解像度を設定します。

[パラメータ:]

名前	型	説明
`maxWidth`	数値	最大表示幅 (ピクセル)。許容される最小値は `0` です。
`maxHeight`	数値	最大表示高さ (ピクセル)。許容される最小値は `0` です。

戻り値:

タイプ

void

setMicrophone (有効化) → {Promise|Promise.<{code: AudioErrorCode、メッセージ: string}>}

マイクを有効または無効にします。

[パラメータ:]

名前	型	説明
`enable`	ブール型	マイクを有効にするには、`true` を指定します。マイクを無効にするには、`false` を指定します。

戻り値:

Promise。拒否された場合、プロミスがエラーオブジェクトを返します。

タイプ

Promise | Promise.<{code: AudioErrorCode、メッセージ: string}>

setMinDisplayResolution(minWidth, minHeight) → {void}

接続に使用する最低表示解像度を設定します。アプリケーションによっては、最低表示解像度が必要になる場合があります。必要な最低解像度がクライアントでサポートされている最高解像度よりも大きい場合は、サイズ変更戦略が使用されます。この関数は慎重に使用してください。サイズ変更戦略により、マウスとタッチ入力システムの精度が低下する可能性があります。

[パラメータ:]

名前	型	説明
`minWidth`	数値	最小表示幅 (ピクセル)。許容される最小値は `0` です。
`minHeight`	数値	最小表示高さ (ピクセル)。許容される最小値は `0` です。

戻り値:

タイプ

void

setUploadBandwidth (値) → {number}

Amazon DCVサーバーへのファイルのアップロードに使用する最大帯域幅を設定します。

[パラメータ:]

名前	型	説明
value	数値	アップストリーム帯域幅の上限 (kbps)。有効範囲は 1024 ~ 102400 kbps です。

戻り値:

- 設定された帯域幅限界。サーバーでファイルストレージ機能が無効になっている場合は null。

タイプ

数値

setVolume (ボリューム) → {void}

オーディオに使用するボリュームレベルを設定します。有効範囲は 0 ~ 100 で、0 が最小ボリューム、100 が最高ボリュームです。

[パラメータ:]

名前	型	説明
volume	数値	使用するボリュームレベル。

戻り値:

タイプ

void

setMicrophone (有効化, deviceId) → {約束|約束.<{code: AudioErrorCode, message: string}>}

〔実験的 - 将来変更される可能性があります] マイクを有効または無効にします。

[パラメータ:]

名前	型	説明
`enable`	ブール型	マイクを有効にするには、`true` を指定します。マイクを無効にするには、`false` を指定します。
`deviceId`	string	マイクのデバイス ID。deviceId を指定しない場合、`default` deviceId が使用されます。

戻り値:

Promise。拒否された場合、プロミスがエラーオブジェクトを返します。

タイプ

Promise | Promise.<{code: AudioErrorCode、メッセージ: string}>

setWebcam (有効化, deviceId) → {約束|約束.<{code: WebcamErrorCode, message: string}>}

ウェブカメラを有効または無効にします。

[パラメータ:]

名前	型	説明
enable	ブール型	ウェブカメラを有効にするには true を指定します。ウェブカメラを無効にするには false を指定します。
deviceId	string	ウェブカメラのデバイス ID。

戻り値:

成功した場合、 がアタッチ/デタッチされたウェブカメラ に解決することを約束しますdeviceId。拒否された場合、プロミスがエラーオブジェクトを返します。

タイプ

Promise.<string> | Promise.<{code: WebcamErrorCode、メッセージ: string}>

syncClipboards() → {ブール値}

ローカルクライアントクリップボードをリモート Amazon DCVサーバークリップボードと同期します。自動コピーがブラウザでサポートされている必要があります。

戻り値:

クリップボードが同期されている場合、関数が true を返します。クリップボードが同期されていない場合、またはブラウザで自動コピーがサポートされていない場合、関数が false を返します。

タイプ

ブール値

認証クラス

dcv モジュールの authenticate メソッドを呼び出して認証トークンを取得する場合は認証クラスを使用する必要があります。使用方法を示した例については「開始方法」セクションを参照してください。

エクスポーズ

- 方法

方法

リスト

- retry() → {void}
- sendCredentials (認証情報) → {void}

retry() → {void}

認証プロセスを再試行します。

戻り値:

タイプ

void

sendCredentials (認証情報) → {void}

クライアントから提供された認証情報を Amazon DCVサーバーに送信します。

[パラメータ:]

名前	型	説明
`credentials`	オブジェクト	提供された認証情報を含むオブジェクト。認証情報の名前と型は、チャレンジで指定された名前と型と同じでなければなりません。

戻り値:

タイプ

void

リソースクラス

リソースクラスは、印刷またはダウンロードされたばかりの対応ファイルを取得または破棄できます。これらのアクションを実行すると、対応するオブザーバー関数 `filePrinted` と `fileDownload` がリソースオブジェクトを唯一の引数としてそれぞれ呼び出されます。これらのリソースを許可または拒否して、参照するファイルを取得または破棄できます。

エクスポーズ

- 方法

方法

リスト

- accept(urlParameters) → {void}
- decline() → {void}

accept(urlParameters) → {void}

リソースをローカルにダウンロードします。

[パラメータ:]

名前	型	説明
`urlParameters`	オブジェクト	リソースを取得するリクエストに渡されたURL検索パラメータのキーと値のペアを含むオプションのオブジェクト。

戻り値:

タイプ

void

decline() → {void}

リソースを破棄します。

戻り値:

タイプ

void

Amazon DCV Web UI SDK

React JavaScript コンポーネントライブラリ。現在、Amazon DCV Server に接続し、リモートストリームとやり取りするためにツールバーをレンダリングDCVViewerする という 1 つの React コンポーネントをエクスポートしています。

エクスポーズ

- コンポーネント

コンポーネント

リスト

- DCVViewer

DCVViewer

リモートストリームとの通信に役立つ機能を備えたツールバーをレンダリングする React コンポーネント。

プロパティ:

リスト

- dcv
- uiConfig

dcv

名前	型	必須	説明
dcv	オブジェクト	可能	Amazon DCV Server への接続を確立するために必要なプロパティを定義し、Amazon DCV Web Client SDKアセットをロードしてDCVリソースにアクセスするURL ログレベルと を設定します。 名前 / 型 / 必須 / 説明 ses / 文字列 / 可能 / Amazon DCV セッション ID。 aut / 文字列 / 可能 / セッションへの接続に

名前	型	必須	説明
			名前 / 型 / 必須 / 説明 使用する認証トークン。 ser / 文字列 / 可能 / 実行中のAmazon DCVサーバーのホスト名とポートを次の形式で

名前	型	必須	説明
			名前 型 必須 説明 指定します。https://dcv_host_address:port. 例: https://my-dcv-server:8443。 bas 文字列 可能 SDKファイルをロードするURL絶対また

名前	型	必須	説明
			名前 型 必須 説明 は相対。 res ase 文字列 No (def "") DCV リソース URL にアクセスする絶対または相対。 onD ect 関数 No (def ( ) => {}) Amazon DCV サーバーから切断す

名前	型	必須	説明
			名前 型 必須 説明 るときに呼び出されるコールバック関数で、接続が閉じられます。 log Logl No (defa Log INF ビューアで使用 l.

名前	型	必須	説明
			名前 型 必須 説明 する ログレベル。

uiConfig

名前	型	必須	説明
`uiConfig`	オブジェクト	No (default: {})	ツールバーを表示するかどうか、および全画面表示ボタンとマルチモニターボタンをツールバーに表示するかどうかを設定するプロパティを定義するオブジェクト。 名前 型 必須 説明 too オブジェクト No (defa {}) ツールバーの設定

名前	型	必須	説明
			名前 型 必須 説明 オプションを定義するオブジェクト。 説明 (default: true) の表示/非表示を定義する

名前	型	必須	説明
			名前 型 必須 説明
			説明 オプション。
			allowMultiScreenMode (default: true) の全画面表示ボタンの表示/非表示を定義するオプ

名前	型	必須	説明
			名前 型 必須 説明
			説明 必須
			ション。
			必須 (default: true)
			のマルチモニターボタンの表示/非表示を定義するオプ

名前	型	必須	説明
			名前 型 必須 説明

説明。

Amazon DCV Web Client のリリースノートとドキュメント履歴 SDK

このページには、Amazon DCV Web Client のリリースノートとドキュメント履歴が表示されます SDK。

トピック

Amazon DCV Web Client SDKリリースノート

このセクションでは、リリース日SDK別の Amazon DCV Web Client のリリースノートを提供します。

トピック

1.8.4 — 2024 年 10 月 1 日

ビルド番号	新機能	変更とバグ修正
• セマンティックバージョン: 1.8.4 • ビルド: 840	次の機能が追加されました。 • 名前を「Amazon DCV Web ClientSDK」に変更 • 高 dpi ディスプレイの新しいAPI enableHighPixel密度を追加 • 互換性のあるブラウザでマイクAPI setMicrophone を選択する実験を追加しました • 新しい接続エラー GATEWAY_BUSY、UNSUPPORTED_CREDENTIAL、TRANSPORT_ を追加ERROR • 新しい終了理由 EXTERNAL_PROTOCOL_CONNECTIONEVICTED、_ を追加 DISCONNECTIONREQUESTED	• ウェブカメラ処理の改善 • オーディオ再生処理の改善 • WebCodecs 処理の改善 • マイクとウェブカメラのプラグとプラグの取り外しを改善 • マルチモニター時のリモートウィンドウのドラッグを改善 • ファイルストレージのアップロードとダウンロードのアクセス許可が正しく伝達されるようになりました • レンダリング時のマイナー修正

Version	リリースノート
• セマンティックバージョン: 1.8.4	変更とバグ修正

Version	リリースノート
• ビルド: 840	•

1.5.10 – 2023 年 12 月 19 日

Version	リリースノート
• セマンティックバージョン: 1.5.10 • ビルド: 684	変更とバグ修正 • ストリームのデコードエラーを修正

1.5.6 — 2023 年 11 月 9 日

Version	リリースノート
• セマンティックバージョン: 1.5.6 • ビルド: 659	変更とバグ修正 • ストリームのデコードとレンダリングのパフォーマンスが向上しました。 • Internet Explorer 11 のサポートが除外されました。

1.4.4 — 2022 年 6 月 29 日

Version	リリースノート
• セマンティックバージョン: 1.4.4	変更とバグ修正 •

Version	リリースノート
• ビルド: 573	ビューワー UI コンポーネントは、それをサポートするブラウザnavigator.keyboard.lock APIの を使用して、全画面で特別なキーを処理するようになりました。 • Chrome 114 以降を使用すると色が不適切になる問題を修正しました。 • WebCodecs 検出が改善されました。 • ウィンドウに入るときのマウスボタンの状態に関する問題を修正しました。 • macOS で修飾キーが押されたままになる問題を修正しました。 • ネットワーク状態が低下したときのオーディオの堅牢性が向上しました。 • メモリリークを修正しました。 • 時間とレベルを含むようにログを改善しました。

1.4.0 — 2023 年 3 月 28 日

Version	リリースノート
• セマンティックバージョン: 1.4.0 • ビルド: 476	新機能 •

Version	リリースノート
	複数のファイルをアップロードする新しい `uploadFiles` メソッドを `FileStorage` オブジェクトに追加しました。 • Viewer UI コンポーネントは、ドラッグアンドドロップでファイルのアップロードを開始できるようになりました。 • WebCodecs ブラウザAPIがオーディオおよびウェブカメラにも使用されるようになりました。 変更とバグ修正 • 同じページから繰り返し接続することによるメモリリークを修正しました。 • `setUploadBandwidth` で 1 Gbps までの値が許可されるようになりました。 • UI コンポーネントのレンダリングを最適化しました。 • Windows でのアニメーションカーソルのサポートが修正されました。 • 同じ操作でテキストデータと画像データの両方が存在する場合のクリップボードサポートの問題を修正しました。 • ウェブカメラ の堅牢性の向上API: リクエストが既に進行中の間は設定を変更できません。`webcam.setEnabled` リクエストのデバイス ID を追跡し、Promise を返しま

Version	リリースノート
	す。Viewer UI コンポーネントは、エラーが発生した場合に通知を表示するようになりました。

1.3.1 — 2022 年 12 月 9 日

Version	リリースノート
• セマンティックバージョン: 1.3.1 • ビルド: 413	変更とバグ修正 • タイムゾーンリダイレクト UI がサーバーと同期しなくなる問題を修正しました。 • 複数回の再接続後のメモリリークを修正しました。 • 切断時に空白ページになる問題を修正しました。 • オーディオデコーダーを終了する際にコンソールに警告が表示されるバグを修正しました。

1.3.0 — 2022 年 11 月 11 日

Version	リリースノート
• セマンティックバージョン: 1.3.0 • ビルド: 407	新機能 • 採用された Cloudscape (https://cloudscape.design) for the UI Viewer component.

Version	リリースノート
	• タイムゾーンリダイレクトのサポートを追加しました。 変更とバグ修正 • DCV ビューワーがフォーカスされている場合の非同期クリップボードの更新の欠落を修正しました。 • クライアント側でディスプレイをスケールする場合、`setDisplayScale` 関数は不要になりました。 • `DCVViewer` コンポーネントは、アンマウントされたときに自動的に `disconnect()` を呼び出すようになりました。

1.2.1 — 2022 年 7 月 21 日

Version	リリースノート
• セマンティックバージョン: 1.2.1 • ビルド: 358	変更とバグ修正 • Amazon DCVサーバー 2019.1 以前への接続に失敗する問題を修正しました。

1.2.0 — 2022 年 6 月 29 日

Version	リリースノート

Version	リリースノート
• セマンティックバージョン: 1.2.0 • ビルド: 352	変更とバグ修正 • 受信したフレームがサポートされている最大解像度 (4096x2160) よりも大きい場合にクラッシュするバグを修正しました。 • リソースオブジェクト (`fileDownload` および `filePrinted` オブザーバーに引数として渡される) には、オブジェクト上で呼び出すことができる `accept` および `decline` メソッドが導入され、それぞれリソースのダウンロードと破棄を実行できるようになりました。 • 切断時の自動クリップボード同期に関するマイナーバグを修正しました。

1.1.3 — 2022 年 5 月 23 日

Version	リリースノート
• セマンティックバージョン: 1.1.3 • ビルド: 329	変更とバグ修正 • オプションを指定するときに接続が成功できない問題を修正しました web-url-path。

1.1.2 — 2022 年 5 月 19 日

Version	リリースノート
•	変更とバグ修正

Version	リリースノート
セマンティックバージョン: 1.1.2 • ビルド: 322	• 接続後に入力が正しく機能しなくなる問題を修正しました。 • スケール比が 1 より大きい場合のマウス座標を修正しました。

1.1.1 — 2022 年 3 月 23 日

Version	リリースノート
• セマンティックバージョン: 1.1.1 • ビルド: 309	変更とバグ修正 • サーバーとの通信がタイムアウトになった時に `Transport Error` を報告します。 • 高解像度のストリーミング時に繰り返し発生するデコードエラーを修正しました。

1.1.0 — 2022 年 2 月 23 日

Version	リリースノート
• セマンティックバージョン: 1.1.0 • ビルド: 295	新機能 • React コンポーネントを使用して Amazon DCV Web `DCVViewer` UI SDKライブラリをリリースします。 • Amazon DCV Web Client を UMDおよび ES モジュールSDKとしてエクスポートします。

Version	リリースノート
	• 高い色精度のサポートを追加しました。 • セッションに接続しているクライアントを一覧表示して操作する機能を追加しました。接続と切断の通知を追加しました。 変更とバグ修正 • WebCodecs のデコードのサポートを改善しました。 • キーボードのさまざまな改善を行いました。 • クリップボードが無効になっているときにセカンドスクリーンを開けないバグを修正しました。

1.0.4 — 2021 年 12 月 20 日

Version	リリースノート
• セマンティックバージョン: 1.0.4 • ビルド: 249	新機能 • 同じページから複数の接続を開けるようになりました。 • SDK からの のロードをサポートします CDN。

1.0.3 — 2021 年 9 月 1 日

Version	リリースノート
• セマンティックバージョン: 1.0.3 • ビルド: 202	新機能 • の実験サポート WebCodecs。これはデフォルトでは無効になっているため、新しいプロパティ `enableWebCodecs` を使用するオブジェクト `ConnectionConfig` を通じて有効にする必要があります。 • クリップボード: Chromium ベースのブラウザのデータ型 `image/png` に対するサポートを追加しました。 • サーバーのスクリーンショットをPNGイメージとして取得するためのオブザーバー/コールバックを追加しました (Amazon DCVサーバー 2021.2 が必要) 。 変更とバグ修正 • キーボード修飾子の処理を改良しました。

1.0.2 — 2021 年 7 月 30 日

Version	リリースノート
• セマンティックバージョン: 1.0.2 • ビルド: 167	• スタイラスイベントの圧力検出を修正しました。 •

Version	リリースノート
	Chrome の韓国語キーボードレイアウトのサポートを改善しました。

1.0.1 — 2021 年 5 月 31 日

Version	リリースノート
• セマンティックバージョン: 1.0.1 • ビルド: 141	• 接続エラーとクローズの理由の伝播を修正しました。 • ファイルストレージチャンクの進行状況の更新を修正しました。 • ウェブカメラの取り扱いを改善しました。 • オーディオイン処理を改善しました。

1.0.0 — 2021 年 3 月 24 日

Version	リリースノート
• セマンティックバージョン: 1.0.0 • ビルド: 81	Amazon DCV Web Client の初回リリースSDK。

ドキュメント履歴

次の表は、Amazon DCV Web Client のこのリリースのドキュメントを示していますSDK。

変更	説明	日付
Amazon DCV Web Client SDK バージョン 1.8.4	Amazon DCV Web Client SDK 1.8.4 が利用可能になりました。詳細については、SDK「v.1.8.4」を参照してください。	2024 年 10 月 1 日
Amazon DCV Web Client SDK バージョン 1.5.6	Amazon DCV Web Client SDK 1.5.6 が利用可能になりました。詳細については、SDK「v.1.5.6」を参照してください。	2023 年 11 月 9 日
Amazon DCV Web Client SDK バージョン 1.4.4	Amazon DCV Web Client SDK 1.4.4 が利用可能になりました。詳細については、SDK「v.1.4.4」を参照してください。	2023 年 6 月 29 日
Amazon DCV Web Client SDK バージョン 1.4.0	Amazon DCV Web Client SDK 1.4.0 が利用可能になりました。詳細については、SDK「v.1.4.0」を参照してください。	2023 年 3 月 28 日
Amazon DCV Web Client SDK バージョン 1.3.1	Amazon DCV Web Client SDK 1.3.1 が利用可能になりました。詳細については、SDK「v.1.3.1」を参照してください。	2022 年 12 月 9 日
Amazon DCV Web Client SDK バージョン 1.3.0	Amazon DCV Web Client SDK 1.3.0 が利用可能になりました。詳細については、SDK「v.1.3.0」を参照してください。	2022 年 11 月 11 日

変更	説明	日付
Amazon DCV Web Client SDK バージョン 1.2.0	Amazon DCV Web Client SDK 1.2.0 が利用可能になりました。詳細については、SDK「v.1.2.0」を参照してください。	2022 年 1 月 29 日
Amazon DCV Web Client SDK バージョン 1.1.0	Amazon DCV Web Client SDK 1.1.0 が利用可能になりました。詳細については、SDK「v.1.1.0」を参照してください。	2022 年 2 月 23 日
Amazon DCV Web Client SDK バージョン 1.0.1	タイプミスを修正しました。小さなバグを修正しました。SDKv.1.0.1 を参照してください。	2021 年 5 月 31 日
初回リリース	このコンテンツの初版です。	2021 年 3 月 24 日

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。