記事
Toshihiko Minamoto · 2021年4月20日 7m read

動画再生時に早送り巻き戻しができない

開発者の皆さん
こんにちは。

先日、RESTのクラスを使って、PDFなどのファイルをアップロード、ダウンロードするサンプルをお送りしたお客さんからご質問を頂きました。
ダウンロード機能を使ってmp4などの動画を表示させた場合、以下のようにvideoの画面下のスライダーを移動しても、再生を進めたり戻したりができませんでした 。

やはり、以下のようにスライダを動かすと、その時点の画像が表示されたほうが良いですね。

そこでgoogleで調べてみると、Http Range request
への対応が必要ということですのでObjectScriptで作成してみました。

表示用html

ブラウザで動画を表示するための簡単なhtmlファイルは以下の通りです。sourceタグのURLにGETメソッドでアクセスし動画データを受信します。

```
<html>
<head>
<title>??????? </title>
</head>
<body>
???????<br>
    <video controls width="800" >
    <source src="http://localhost:52773/csp/storage/River.mp4" type="video/mp4">
    </video>
</body>
</html>
```

RESTクラス

RESTクラス(REST.StreamTransfer)は%CSP.RESTクラスを継承しており、管理ポータルのウェブアプリケーション設定により/csp/storageへのアクセスがあると、このRESTクラスがハンドリングするようになっています。

REST.StreamTransferクラスには以下のURLMapを持っており、ブラウザの表示用HTMLからGETメソッドでのアクセスがあると、/csp/storage配下のRiver.mp4をパラメータとしてDownloadメソッドが呼び出されます。

```
XData UrlMap [ XMLNamespace = "http://www.intersystems.com/urlmap" ]
{
<Routes>
<Route Url="/" Method="GET" Call="ListFile" Cors="true"/>
<Route Url="/:filename" Method="POST" Call="Upload" Cors="true"/>
```

```
<Route Url="/:filename" Method="GET" Call="Download" Cors="true"/>
</Routes>
}
```

呼び出されるDownloadメソッドは以下の通りです。渡されたRiver.mp4はfilenameパラメータとなります。

```
ClassMethod Download(filename As %String)
{
    set tSC=$$$OK
    try {
        // ??????????
        set file=##class(%File).NormalizeDirectory(..#Storage)_filename
        if '##class(%File).Exists(file) {
            set %response.Status="404 File not found"
            quit
        }
        set %response.Expires=120

       // ContentType???

     s
et filetype=
$select(filename[".":$zcvt($piece(filename,".",*),"L"),1:"") // ???(???)???
        if filetype'="" {
            set %response.ContentType=$get(^FileType(filetype))  /// ??????????
        }
        set:%response.ContentType="" %response.ContentType="application/binary"

        // Accept-Ranges???
        do %response.SetHeader("Accept-Ranges","bytes")

        //????????????
        set stream=##class(%FileBinaryStream).%New()
        set stream.Filename=file

        // ????(Range???)?????????
        set range=%request.GetCgiEnv("HTTP_RANGE")
        if range'="" {
            set unit=$piece(range,"="),range=$piece(range,"=",2)
            // ??????????
            set range=$zstrip($piece(range,","),"<>W")
        } else {
            set range="0-"
        }
        if range="0-" {
            // ????????????
            set %response.Status="206 Partial Content"
            do %response.SetHeader("Content-
Range","bytes 0-"_(stream.Size-1)_"/"_stream.Size)

            // ??????
            set tSC=stream.OutputToDevice()
        } else {
            // ???????????
            set start=$zstrip($piece(range,"-"),"<>W"),end=$zstrip($piece(range,"-",2
),"<>W")
            set:end="" end=stream.Size-1
            set len=end-start+1
```

```
            // ????????
            do %response.SetHeader("Content-
Range","bytes "_start_"-"_end_"/"_stream.Size)
            set %response.Status="206 Partial Content"

            // ???????
            set tSC=stream.OutputToDeviceAt(start+1,.len)
        }

    } catch {
        set %response.Status="500 Download error"
    }
    quit $$$OK
}
```

このメソッドではStorageパラメータで指定されたディレクトリに対して、与えられたパラメータと同じ名称のファイルをオープンし、その内容をHTTPレスポンスとしてクライアントに返す処理を行っています。

今回、Http Range Requestを実現するために以下の処理を追加しています。

1. Accept-Rangesヘッダの追加
   HTTPレスポンスを返す際に、このヘッダを追加することでクライアントに対してRange requestに対応していることを示しています。

```
            // Accept-Ranges???
            do %response.SetHeader("Accept-Ranges","bytes")
```

2. Rangeヘッダの読み込み
   環境変数HTTPRANGEでヘッダの内容を読み込みます。Rangeヘッダのフォーマットは Range: <unit>=<range-start>-<range-end> となり、<unit>には通常bytesが入り、<range-start>は開始バイト数、<range-end>は終了バイト数が入ります。 Rangeヘッダは以下の部分で読み込み、開始バイト数と終了バイト数を求めています。規格としてはカンマ区切りで複数の範囲を要求することも可能ですが、今回は省略しています。

```
            // ????(Range???)?????????
            set range=%request.GetCgiEnv("HTTP_RANGE")
            if range'="" {
                set unit=$piece(range,"="),range=$piece(range,"=",2)
                // ??????????
                set range=$zstrip($piece(range,","),"<>W")
            } else {
                set range="0-"
            }
```

3. Content-Rangeタグ、ステータスコードの設定 2で取得した範囲がストリーム全体でなかった場合、出力する範囲をContent-Rangeヘッダに設定、ステータスコードを206を設定しています。 Content-Rangeヘッダのフォーマットは以下の通りです。 Content-Range: <unit> <range-start>-<range-end>/<size> <unit>はbytesとし、取得した範囲に<range-end>が指定されていなかった場合、ストリームサイズ-1を設定しています。

```
        // ???????????
      set start=$zstrip($piece(range,"-"),"<>W"),end=$zstrip($piece(range,"-",2),"<
>W")
        set:end="" end=stream.Size-1
        set len=end-start+1

      // ????????
```

```
   do %response.SetHeader("Content-
Range","bytes "_start_"-"_end_"/"_stream.Size)
   set %response.Status="206 Partial Content"
```

4. 指定範囲の出力 2.で取得した範囲がストリーム全体でなかった場合、OutputToDeviceAtメソッドを使用して一部分を出力しています。このメソッドのパラメータは開始位置と出力文字数ですが、Http Range requestが0から始まるのに対しOutputToDeviceAtは1から始まりますので、start+1としています。

```
   // ???????
   set tSC=stream.OutputToDeviceAt(start+1,.len)
```

以上です。

github にソースコードを入れておきますので、ご興味がありましたら、ご参照ください。

また、ご意見やご質問等ありましたら、お気軽に返信いただければと思います。

#REST API #Caché #InterSystems IRIS

ソースURL:
https://jp.community.intersystems.com/post/%E5%8B%95%E7%94%BB%E5%86%8D%E7%94%9F%E6%99%82%E3%81%AB%E6%97%A9%E9%80%81%E3%82%8A%E5%B7%BB%E3%81%8D%E6%88%BB%E3%81%97%E3%81%8C%E3%81%A7%E3%81%8D%E3%81%AA%E3%81%84