Vol.2018-GN-103 No.10
Vol.2018-CDS-21 No.10
Vol.2018-DCC-18 No.10
2018/1/26

動画のながら視聴に特化したウェブブラウザの開発に関する研究

井上弘仲

概要： 近年、ウェブブラウザで動画を視聴することは一般的となった。その際、テレビの垂れ流し視聴と同様に、他の Web ページと同時に動画を閲覧する“ながら視聴”は需要として考えられる。本研究では、既存のウェブブラウザと比べて、容易なユーザ操作で“ながら視聴”を行えるシステムを開発する。本システム独自の“ながら視聴”機能では、大手動画視聴サイトにアクセスし、その中から動画情報を別ウィンドウに取り出してユーザ任意のサイズ・レイアウトで閲覧することができる。本システムには Youtube 以外の大手動画視聴サイトへの対応や、複数動画の同時ストックなどの独自性がある。しかし既存のブラウザと比べ基本的なブラウジング機能が劣ること、また他のユーザによる検証を行っていないという問題点があるため、今後解決してシステムの向上を図り製品化を目指していく。

キーワード：ソフトウェア構築、ウェブブラウザ，動画，ながら視聴 [**]

1. はじめに

1.1 背景

近年、YouTube やニコニコ動画といった動画視聴サイトの出現と普及により、ウェブブラウザで動画を視聴することは一般的となった。映画館で見る映画などとは違い、プライベートな空間内での動画視聴は、それだけに集中して視聴する必要もなく、テレビにおける垂れ流し視聴などと同様に、他のことをやりながら視聴するスタイルが考えられる。PC 上で視聴する際、動画の流れる場所だけ表示すれば、それ以外の場所で、他の Web サイトの閲覧や word による書類作成など、他の作業を行うことができるスペースができる。

以上のことから、Web サイト上の動画を見ながら他のソフトウェアも同時に起動・作業を行う“ながら視聴”（図 1）は需要として考えられる。

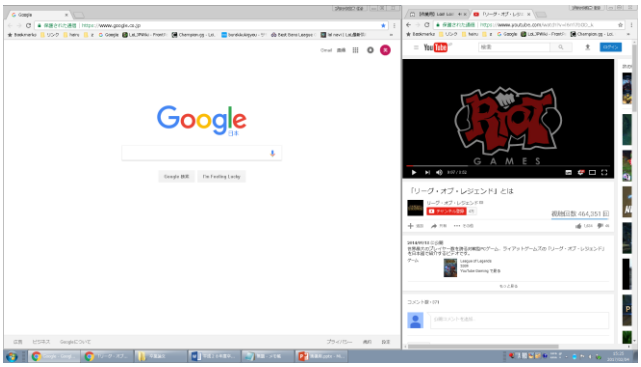


図 1　既存ウェブブラウザでながら視聴（他の Web サイトと動画視聴を同時に行う）を行う例

1.2 目的

図 1 を例とすると、現存のウェブブラウザでながら視聴を行う場合、以下のような問題点がある。

- 動画視聴ウィンドウと他 Web サイトの閲覧のウィンドウを分ける作業が毎回必要。
- 動画領域に合わせた動画視聴用ウィンドウの位置とサイズの調整が毎回必要

本研究ではこれらの問題を解決し、ユーザが容易にながら視聴を実現できるソフトウェアを開発する。

2. 研究手法

2.1 事前調査

ながら視聴が可能なソフトウェアを開発することにあたって、まず複数コンテンツを同時視聴する機能を持つ現存のシステムについての調査を行った。

調査

- **既存システム 1**

Floating YouTub

- **概要**

Floating YouTub は Google Chrome のアドオンである。
このアドオンは Google Chrome のツールバーにボタンが追加される。YouTube の動画再生の画面で追加されたボタンを押すことで、別のウィンドウが開き、そこに動画が表示される。そのウィンドウのサイズ、配置を自由にできる。

- **問題点**

複数の動画を同時に視聴、取得することはできない。YouTube の動画しか取得できない。Google Chrome でしか使用できない。

- **既存システム 2**

複数デバイスによるマルチコンテンツ閲覧のためのコンテンツ配信・表示制御手法*[1]

- **概要**

この研究では、複数のデバイス（PC、携帯電話）を用いてマルチコンテンツを閲覧するための、コンテンツの配信・表示制御の仕組みを考えた。
複数のデバイスでコンテンツの配信を行った際、個々のデバイスの方法で表示される為、同じコンテンツでも異なる表示がされる。だから、配信表示の統一化するシステムを提案する。

- **問題点**

提案の段階でソフトウェア開発には至っていない。

- **既存システム 4**

ゲーム実況動画における動画多画面視聴支援システム提案*[2]

- **概要**

この研究では、1 つの Web ブラウザ上で、動画投稿サイト（You Tube、ニコニコ動画など）に投稿された動画、配信

Vol.2018-GN-103 No.10
Vol.2018-CDS-21 No.10
Vol.2018-DCC-18 No.10
2018/1/26

動画（Ustream tv、ニコニコ生放送など）を複数同時に視聴することのできる Web アプリケーションを提案する。

- **問題点**

複数同時に視聴する際、投稿された動画は自分でアップロードしないといけない。

考察

以上の調査結果から、本研究では一つのデバイスで、アップロード等の事前準備が必要なく動作する、様々な動画サイトの動画を複数視聴でき、既存のブラウザに依存しない、ソフトウェアの開発が必要であると考えられる。

- アドオンではなくなぜソフトゥエアの開発なのか。

アドオンには、ブラウザ同士の互換性が無い場合がある、ブラウザの仕様が変わると使用できない、拡張性がブラウザによって左右される。など既存のブラウザに依存するところが多いことからソフトゥエアの開発を行う。

2.2 本研究システムの要件

2.1 の調査結果から、本研究において開発するながら視聴システムは事前準備の必要がなく、大手動画視聴サイト（You Tube、ニコニコ動画など）から動画のみを取得し、容易に閲覧する機能が必要である。したがって、大手動画視聴サイトなどを通常のブラウザと同程度の機能をもって閲覧できる「ウェブブラウザの基本機能」、またそこから動画情報を取得して他のソフトウェアと同時に閲覧するための「ながら視聴の機能」が必要である。

以下は、本研究ソフトウェアの要件を大きく上記 2 つの機能で分けて列挙したものである。

- **ウェブブラウザの基本機能**
- 戻る、進む、URL 移動などの基本機能
- タブ機能

ながら視聴の機能

- 動画の取得機能
- 取得した動画のサイズ制御、削除機能
- レイアウトの保存・呼び出し・削除機能

2.3 ソフトウェア設計

表１　ベース・ブラウザの機能一覧

基本機能
「戻る」「進む」「更新」「中止」「ホーム」「URL」 「「URL」のテキストボックスの能動的サイズ変化」 「閉じた際のレイアウトの保存」
タブ機能
「新しいタブを追加」「タブ選択」「タブを削除」 「アクティブなタブの制御」

表２　ながら視聴の機能一覧

動画の取得機能
「ムービービューアーの生成」「動画の取得」 「ムービービューアー内の動画数表示」 「ムービービューアーの単一表示」
取得した動画の制御、削除機能
「取得した動画の削除」：個別 「取得した動画の削除」：全部 「取得した動画のサイズ変更」
レイアウトの保存・呼び出し機能
「レイアウトの保存」 「レイアウトの呼び出し」 「保存したレイアウトの削除」

本研究ソフトウェアは、2.2 の「ウェブブラウザの基本機能」「ながら視聴の機能」の要件を満たさなければならない。表 1 および表 2 は、要件から更に分けたソフトウェアの詳細設計一覧である。

2.4 開発手順

本研究ソフトウェアは、以下１）～４）手順で開発した。

(1) ベース・ブラウザの開発

表 1 の機能を持つソフトウェアを開発する。このソフトウェアを以下“ベース・ブラウザ“と呼ぶ。そしてベース・ブラウザに、以下のながら視聴の機能を追加する。

(2) 動画の取得機能の開発

動画取得して表示する為の動画視聴用ウィンドウの生成。以下、このウィンドウのことを“ムービービューアー”と呼ぶ。このムービービューアーに、容易に動画を取得できる機能の開発。

(3) 取得した動画の制御の開発

ムービービューアー内に２）で取得した動画のサイズを変更できる機能の開発。

ムービービューアー内に２）で取得した動画を削除できる機能の開発。

(4) 複数ウィンドウのサイズレイアウトの保存・呼び出し・削除機機能の開発

ベース・ブラウザ、ムービービューアーのサイズレイアウトを保存できる機能の開発。

保存したサイズレイアウトの呼び出し機能の開発。

保存したサイズレイアウトを削除できる機能の開発。

3. ソフトウェアの構築

3.1 開発環境

2.研究手法で示したように、本研究ソフトウェアはウェブブラウザとしての機能だけではなく、ウィンドウレイアウト制御など、OS の基本機能とも密接に連携し、任意の

Vol.2018-GN-103 No.10
Vol.2018-CDS-21 No.10
Vol.2018-DCC-18 No.10
2018/1/26

拡張が可能な開発環境が必要である。そのような開発環境として代表的なものは２つあり、以下は情報取得のしやすさに着目してそれぞれの特徴を列挙したものである。

(1) **Web browser コンポーネント（Internet Explorer の素となっている）**

- 機能詳細に関する Microsoft 社の公式サイトがある
- 新しい情報と古い情報が混ざっていることがある
- 個人サイト含め情報量が多い

(2) **WebKit（元 Google Chrome、Safari の素になっている）**

- (1) のような機能詳細のサイトが見つからない
- 一部の機能に重点を置いて紹介する個人サイトは散見される
- 基本的に英語であり日本語の情報量は少ない

本研究では、機能詳細の公式サイトがあり情報量が多いことから Web browser コンポーネントを使用した。

(3) **開発言語、開発ソフトウェア**

開発言語は「C#」を使用し、開発ソフトウェアとして「Visual Studio 2015(Windows Form Application)」を使用した。

3.2 Windows From Application 内で使用したツール一覧

以下は開発に当たって Windows From Application 内で使用したツールの一覧と概要、また図２は各ツールのグラフィックである。

- **Form**

アプリケーションに表示されるすべてのウィンドウの表現。

- **Web browser**

Form 内の Web ページをユーザが参照できるようにする。

- **Tool Strip**

ツールバー。および多数の表示オプションやオーバーフローおよび実行時の並び替えなどをサポートするその他のツール要素を提供。

- **Tool Strip Label**

Tool Strip 内で使用できる Label、コントロールの実行時の情報または説明用のテキストを提供。

- **Tool Strip Button**

Tool Strip 内で使用できる Button、ユーザがクリックしたときにイベントを発生させる。

- **Tool Strip Split Button**

Tool Strip 内で使用できる Button。通常の Button の右側にドロップダウンボタン（▼下矢印ボタン）が付いている。ドロップダウンボタンをクリックすると、ドロップダウンメニューが表示される。

- **Tool Strip Combo Box**

Tool Strip 内で使用できる Combo Box。使用できる値のドロップダウンリストを含む、編集可能なテキストボックスを表示する。

- **Tool Strip Menu Item**

Tool Strip Combo Box の▼を押すことで表示される一覧。

- **Tool Strip Separator**

Tool Strip 内で使用できる Separator、ツールの区切りを示す縦棒。

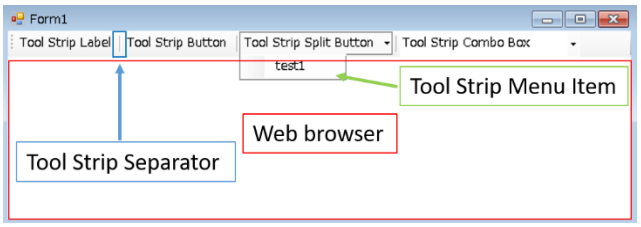


図２　Windows From Application 内で使用したツールグラフィック

3.3 ベース・ブラウザ機能の構築

表 1 で提示したソフトウェア設計に従い、ベース・ブラウザ機能の構築を行った。機能は大きく「基本機能」「タブ機能」に分けられる。

- **基本機能**

以下は「基本機能」のユーザインタフェースと実装機能の説明に分けて記述する。

- **ユーザインタフェース**

図 3 はベース･ブラウザのユーザインタフェースである。「基本機能」に関してはボタンやテキストボックス等の配置などを既存のブラウザと違和感がないよう酷似させている（図 3、図 4 赤線部）。これは、同じような使用感にすることで、ユーザが本ソフトウェアを容易に導入できるようにするためである。基本機能の配置は、ウィンドウ内の最上部、左から順に「戻る」「進む」「更新」「中止」「動画の取得」「配置を保存する」「URL」となっている。（図 3 赤線部）

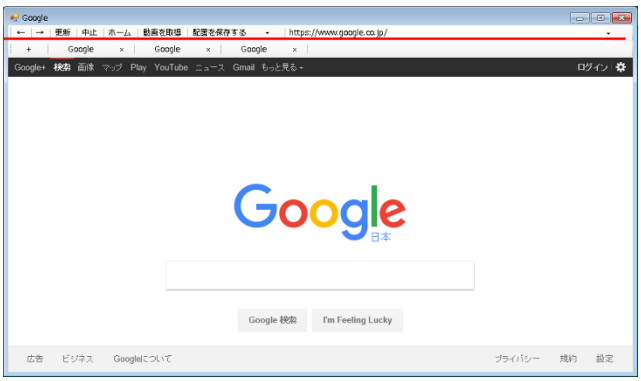


図３　ベース・ブラウザ

Vol.2018-GN-103 No.10
Vol.2018-CDS-21 No.10
Vol.2018-DCC-18 No.10
2018/1/26

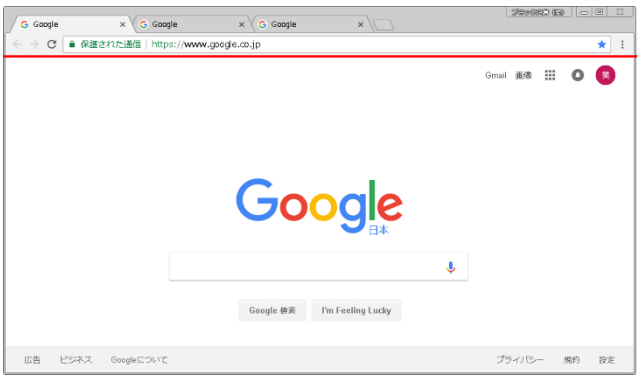


図 4　Google Chrome

図 5　ベース・ブラウザ

● 「戻る」

「戻る」の場所を押すことで 1 つ前の Web サイトに戻ることができる。

● 「進む」

「進む」の場所を押すことで「戻る」を使用しているとき戻る前の Web サイトに戻ることができる。

● 「更新」

「更新」の場所を押すことで表示している Web サイトを再度読み込むことができる。

● 「中止」

「中止」の場所を押すことで表示している Web サイトの読み込みを止めることができる。

● 「ホーム」

「ホーム」の場所を押すことで設定しているホームページに移動できる。

● 「URL」

「URL」に使用しているテキストボックスには、URL を直接入力でき、Enter キーを押すことで、入力された URL の Web サイトに移動できる。

● **タブ機能**

以下は「タブ機能」のユーザインタフェースと実装機能の説明に分けて記述する。

● **ユーザインタフェース**

「タブ機能」のユーザインタフェースについても、基本機能の構築と同じ理由で酷似させている（図 5、図 6 赤線部）。タブ機能の配置はウィンドウ内の最上部から 2 段目、左から順に「新しいタブを追加」「タブ選択」「タブを削除」となっている（図 5 赤線部）。

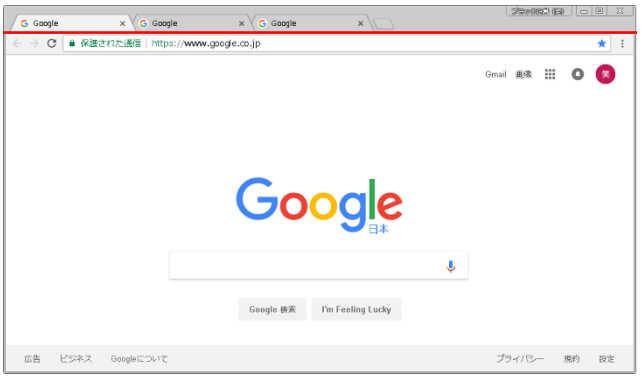


図 6　Google Chrome

● 「新しいタブを追加」

「＋」の場所（図 5 黄の枠）を押すことで新しいタブを生成する。

● 「タブ選択」

「タブ」の場所（図 5 緑の枠）を押すこと、そのタブが保持する Web ページを表示する。

● 「タブ削除」

「タブ上の×ボタン」の場所（図 5 青の枠）を押すことで、そのタブを削除する。

3.4 ながら視聴機能の構築

表 2 で提示したソフトウェア設計に従い、ながら視聴機能の構築を行った。機能は大きく「動画の取得機能」「取得した動画の制御、削除機能」「レイアウトの保存・呼び出し機能」に分けられる。

(1) 動画の取得機能

以下は動画の取得機能のユーザインタフェースと実装機能の詳細に分けて記述する。

● **ユーザインタフェース**

図 7 は「動画の取得機能」のユーザインタフェースであり、右のウィンドウが、動画取得して表示する為の動画視聴用ウィンドウ“ムービービューアー”である。

Vol.2018-GN-103 No.10
Vol.2018-CDS-21 No.10
Vol.2018-DCC-18 No.10
2018/1/26

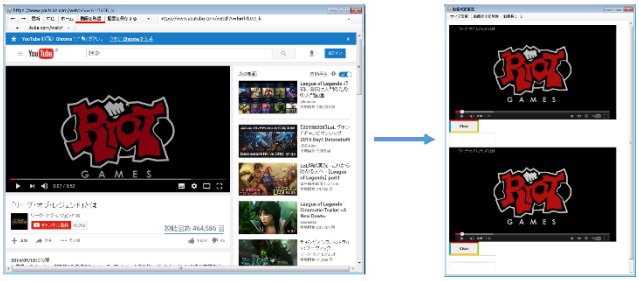


図 7　動画の取得

「動画を取得」（図 7 左赤線部分）を押すことムービービューアーが表示される。「動画を取得」を押したときに YouTube の動画視聴サイトであった場合、表示している動画をムービービューアーに取得し、配置する。

ムービービューアー内に表示される取得動画数は、「動画数」（図 8 右赤枠部分）に常に表示されている。

ムービービューアーは任意のタイミングで、ウィンドウ右上部の「×」で閉じることができる。

ムービービューアーは最大で 1 つしか表示されないように制御されており、既に表示されているときに「動画を取得」を押しても、新しくムービービューアーは作られない。

- **実装機能の詳細**

「ムービービューアーの生成」

ムービービューアー内の表示コンテンツは、プログラム側にて HTML を生成・変化させている。HTML ソースは以下のような構造になっている

```
<html>
    <head>
```

ムービービューアの見た目調整用 css

```
    </head>
    <body>
```

動画 1 の HTML

```
[動画]
<iframe class = douga id = abc + “1” + width = 560 height = 315 src = https://www.youtube.com/embed/ + douga + frameborder = 0 allowfullscreen >
</iframe>
[Close ボタン]
<a class= btn id = tozi + “1” + href = #>Close</a><br/>
[動画サイズ変更用のテキストボックス]
<input id = wid" + “1” + type='text' name='userName'/><br/>
<input id = hei" + “1” +type='text' name='userName'/><br/>
```

動画 2 の HTML

```
[動画]
<iframe class = douga id = abc + “2” + width = 560 height = 315 src = https://www.youtube.com/embed/ + douga + frameborder = 0 allowfullscreen >
</iframe>
[Close ボタン]
<a class= btn id = tozi + “2” + href = #>Close</a><br/>
[動画サイズ変更用のテキストボックス]
<input id = wid" + “2” + type='text' name='userName'/><br/>
<input id = hei" + “2” +type='text' name='userName'/><br/>
```

⋮

以下、動画の数だけ繰り返す

```
    </body>
</html>
```

ムービービューアー内の動画表示は、ブログなどで使用される iframe タグによる動画の埋め込み機能を使用した。

「動画の取得」

動画の取得ボタンには Tool Strip Button を使用した。Click イベントハンドラが起動すると、一番前に表示している Web サイトの URL を取得する。取得した URL によって視聴サイトの種類や個々の動画を判別する。例えば YouTube のサイトの URL には「YouTube」という文字列が入っており、動画視聴ページではさらに「v=」という文字列が入っている。その 2 つが URL にあるか判断し、入っていれば動画の取得処理に入る。

動画をひとつ取得するたびに「「動画」「Close ボタン」「テキストボックス」を持つ以下の HTML が追加される。

```
<iframe class = douga id = abc + “▲” + width = 560 height = 315 src = https://www.youtube.com/embed/ + douga + frameborder = 0 allowfullscreen >
</iframe>
[Close ボタン]
<a class= btn id = tozi + “▲” + href = #>Close</a><br/>
[動画サイズ変更用のテキストボックス]
<input id = wid" + “▲” + type='text' name='userName'/><br/>
<input id = hei" + “▲” +type='text' name='userName'/><br/>
```

上記 HTML ソース内の▲部分には、これまでムービービューアーに取得した累計動画数のテキストが挿入される。

上記の HTML は文字列として変数に入れている、それにより、この HTML を 1 つの塊として、「動画を取得」を押すたびに、加えることで、動画を連続して表示している。

2 行目の「douga」の変数には URL から取得した各動画固有の文字列が入っている、これにより、動画サイトのどの動画か指定して表示している。

▲部分に累計動画数を入れることによって、「動画」「Close ボタン」「テキストボックス」はそれぞれユニークな id 名を持つ。これにより(2)で後述する取得した動画の制御を行えるようにしている。

「ムービービューアーの動画数表示」

「動画数」の表示は、数値部分は Tool Strip Label で表示を行った。「動画の取得」を行った際には、表示する数値の

Vol.2018-GN-103 No.10
Vol.2018-CDS-21 No.10
Vol.2018-DCC-18 No.10
2018/1/26

方に足し、(2)で後述する「取得した動画の削除」を行った際には数値を減らす動作をする。

「ムービービューアーの単一表示」

「動画を取得」を押した時、ムービービューアーを表示させる処理をした際、表示だけの処理では、複数ウィンドウが表示されるそのため、表示を制御する為の変数を用意した。その変数は bool 変数（true、false の値を扱える、スイッチの様な変数）を使用している。ムービービューアーが表示しているかの判断はムービービューアーのタイトルの値を使用している、表示されているときは値が入っているので、true を返す。表示されてないときには、タイトルの値に null が入っているので、false を返す。それにより、bool 変数が true なら、「動画を取得」が押されたときに、表示する処理を省く。そして、false なら、表示の処理を行う。

(2) 取得した動画の制御、削除機能

取得した動画の制御機能は、大きく「取得した動画の削除」「取得した動画のサイズ変更」の機能に分けられる。「取得した動画の削除」はさらに個別削除と全削除に分けられる。以下、各機能のユーザインタフェースと機能実装の詳細に分けて記述する。

- **ユーザインタフェース**

「取得した動画の削除」：個別

動画横、下にある「Close」を押すことで、その「動画」「Close ボタン」「テキストボックス」を削除することができる。下記の画像は動作したものである。

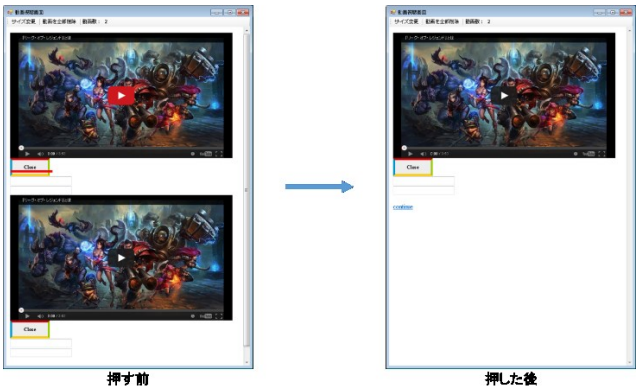


図 8　取得した動画の削除、個別

「取得した動画の削除」：全部

ムービービューアー内の「動画の全削除」を押すことですべての「動画」「Close ボタン」「テキストボックス」を削除する。下記の画像は動作したものである。

図 9　取得した動画の削除、全部

「取得した動画のサイズ変更」

動画下の「テキストボックス」に数値を入力して「サイズ変更」を押すことで動画のサイズを変更する。下記の画像はサイズ変更を行ったものである。

図 10　取得した動画のサイズ変更

- **実装機能の詳細**

「取得した動画の削除」：個別

ムービービューアー上でマウスを左クリックしたときに動作する「Document .Mouse Down イベントハンドラ」において、クリックした場所の HTML 要素を取得する処理をする。取得した HTML 要素が「Close ボタン」であるかを判断し、「Close ボタン」の場合には、その id 名のテキスト情報からユニークな番号（▲部分）を取得する。同じユニーク番号 id を持つ「動画」「Close ボタン」「テキストボックス」の HTML を削除することで、動画を個別に削除することができる。

「取得した動画の削除」：全部

Tool Strip Button を使用した。Click イベントハンドラが起動すると、ムービービューアー上の HTML を空白にする動作をする。

「取得した動画のサイズ変更」

Tool Strip Button を使用した。Click イベントハンドラが起動すると、ムービービューアー上の HTML を取得し、その中の「動画サイズ変更用テキストボックス」に入っている数値を取得する。その後テキストボックスに割り振られたユニーク番号と一致する「動画」要素について、width および height プロパティを取得した数値で適用を行う。

(3) レイアウトの保存・呼び出し・削除機能

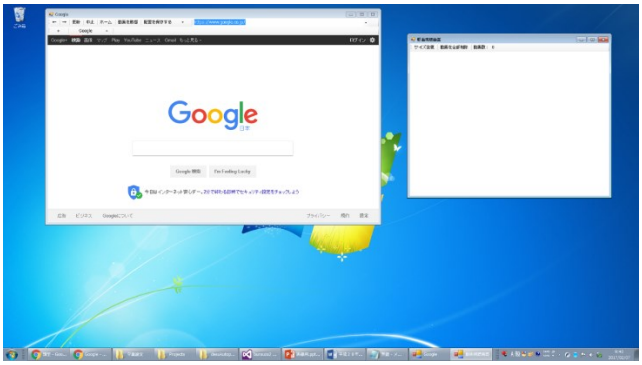
レイアウトの保存・呼び出し機能は大きく「レイアウトの保存」「レイアウトの呼び出し」「保存したレイアウトの

Vol.2018-GN-103 No.10
Vol.2018-CDS-21 No.10
Vol.2018-DCC-18 No.10
2018/1/26

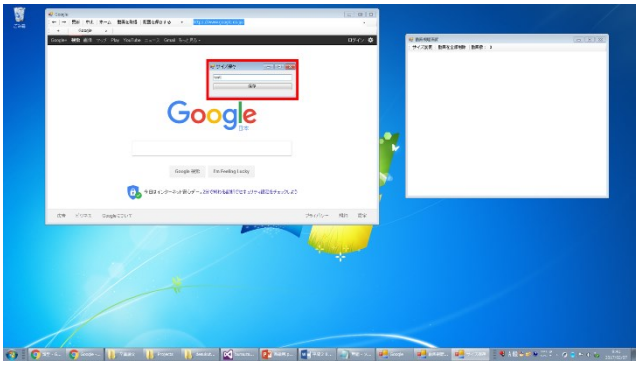
呼び出し」の３つに分かれるが、ユーザ操作が煩雑である、なので一連の流れを図と共に各インタフェースを外観する。その後、実装機能の詳細に関して記述する。

- **ユーザインタフェース**

＜保存の流れ＞



手順１：各ウィンドウを好きなサイズ、配置にする。



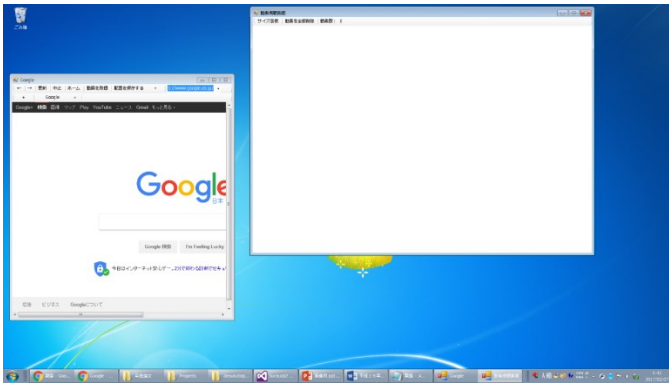
手順２：「配置を保存する」ボタンを押すとダイアログが出て、この配置を「test」と名前をつけ保存する。

「レイアウトの保存」

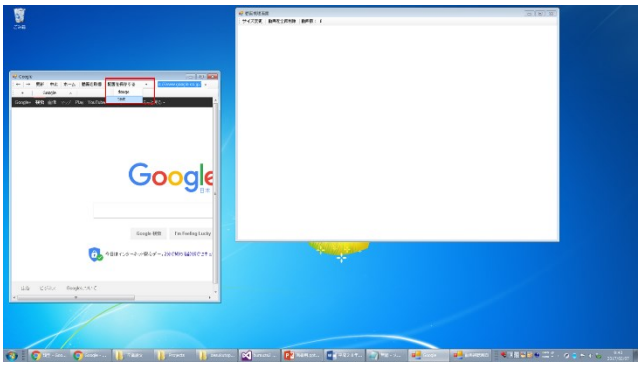
ベース・ブラウザの「配置を保存する」ボタンを押すことでウィンドウ「サイズ保存」（手順２赤枠）が呼び出される。

「サイズ保存」には入力できるテキストボックスがあり、保存するレイアウトの名前を入力して、「保存」を押すことで現在のベース・ブラウザとムービービューアーのレイアウト情報に名前を付けて保存できる。

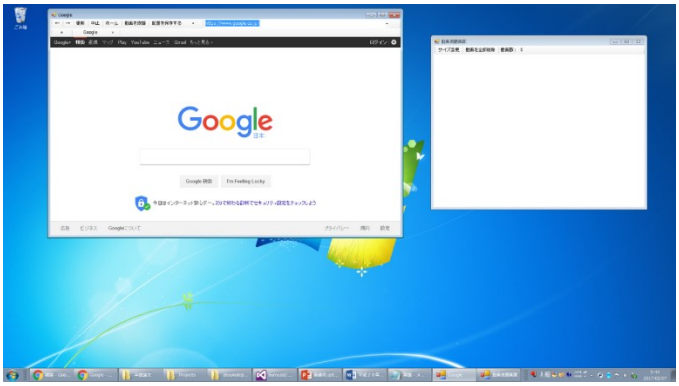
＜呼び出しの流れ＞



手順１：サイズ、配置を変える。



手順２：「配置を保存する」ボタンの隣の▼を押すと、先ほど保存した「test」があるので、クリックする。

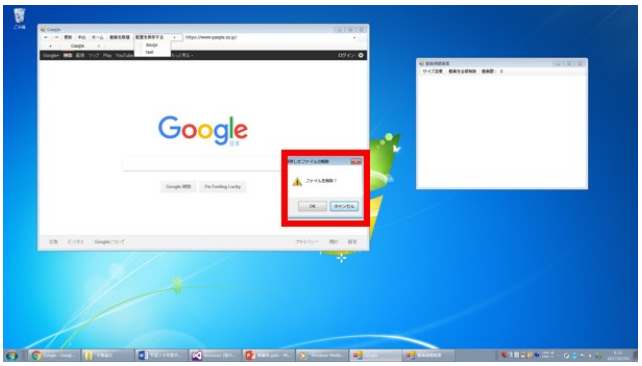


手順３：「test」で保存したサイズ、配置に変更される。

「レイアウトの呼び出し」

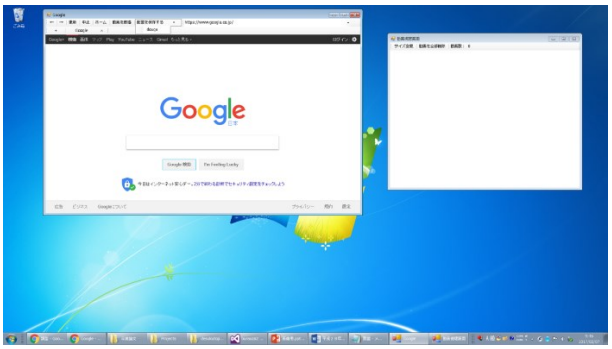
「配置を保存する」ボタンの右の▼を押すことで「レイアウトの保存」で保存した名前が表示される（手順２赤枠）。それを押すことで保存したレイアウトを呼び出し適応する。

＜削除の流れ＞



手順１：「配置を保存する」ボタンの隣の▼を押して、「test」を右クリックするとダイアルログが出る。

Vol.2018-GN-103 No.10
Vol.2018-CDS-21 No.10
Vol.2018-DCC-18 No.10
2018/1/26



手順２：「OK」を押すことで保存した「test」が削除される。

「保存したレイアウトの削除」

「配置を保存する」ボタンの右の▼を押し、保存したレイアウト一覧表示させる。レイアウト名を右クリックすることでダイアログが出現（手順 1 赤枠）、OK を押すことで保存したレイアウトを削除することができる。

- **実装機能の詳細**

「レイアウトの保存」

レイアウト保存ボタンには Tool Strip Split Button の左側のボタンを使用した。Click イベントハンドラが起動すると「サイズ保存」ダイアログを表示する。ダイアログ中で保存が押されたら、「テキストボックスに入力された文字列.txt」というテキストファイルを生成する。さらにテキストファイルの内容に、そのとき表示しているベース・ブラウザとムービービューアーの幅と高さ、またディスプレイ上の左上からの相対位置をピクセル単位の数値として書き込み、保存する動作をする。

「レイアウトの呼び出し」

▼を押されたときに表示される一覧には Tool Strip Menu Item を使用した。まずレイアウト保存用のテキストファイルを全て読み込み、そのファイル名を一覧に表示する。ユーザに選択されて各項目の Click イベントハンドラが起動すると、一覧で押されたものと同じ名前のテキストファイルを探して内容を読み取り、幅と高さ、相対位置として書くウィンドウプロパティに適応する。

「保存したレイアウトの削除」

▼を押されたときに表示される一覧には Tool Strip Menu Item を使用している。Click イベントハンドラ中でクリックボタンの判断を行い、右クリックの場合には削除ダイアルログを表示する。OK が押されると、選択した名前と一致するテキストファイルを削除し、同時に Tool Strip Menu Item の一覧のデータからも削除する動作をする。

4. 結論と今後

4.1 結論

本研究では、動画エリアのみを抽出して表示することで、１つのディスプレイ内で動画を視聴しながら他のスペースで作業をする、“ながら視聴” を容易に行うことができるソフトウェアの開発を行った。

本研究システムは、通常のブラウザの基本機能を持つベース・ブラウザと、動画視聴用ウィンドウであるムービービューアーを持ち、動画の取得、サイズ変更、レイアウトの保存のながら視聴機能を実装した。本研究システムは既存のシステムと比べて事前の動画アップロードなどの事前準備を必要とせず、動画視聴サイトから直接動画を取得することができる。また複数動画のストックという独自性も持つ。

4.2 現状の問題、今後の展開

現状の本研究ソフトウェアの問題点を以下に列挙する。

- まだ基本的なウェブブラウザとしての機能が既製品（Internet Explorer Google Chrome など）に比べ劣るので基本機能を充実させる。
- 現在、動画視聴用ウィンドウ内の動画サイズを変更するには、縦幅横幅をテキストとして入力し、サイズ変更ボタンを押す必要がある。これは、操作として煩雑であり、ユーザが求めるサイズに調整するのが難しい。そこで、動画視聴ページのサイズ変更をもっと感覚的に変えられるようにする。（マウスによるドラックなどを検討中）
- 現在ソフトウェア開発の段階で、他のユーザによる使用感の検証を行っていないので、実際に使いやすいのか分からない。他のユーザに使用してもらう、ネットに公開するなどを検討している。

このような問題を解決して今後製品化していきたい。

参考文献

[1]”複数デバイスによるマルチコンテンツ閲覧のためのコンテンツ配信・表示制御手法”,
著者,赤星祐平,木俵豊,田中克己, 収録されている論,文集,第 16 回データ工学ワークショップ(DEWS2005)論文集, 発行日,2005-03
[2]”ゲーム実況動画における動画多画面視聴支援システム提案”,
著者,中野裕太,服部哲,速水治夫,収録されている論文集,マルチメディア、分散協調とモバイルシンポジウム 2011 論文集, 収録ページ pp.370-373, 発行日 2011-06-30