

# DOMinGO : 対話的な Web ページ表示制御システム

早川 匠<sup>1</sup> 増井 俊之<sup>2</sup>

**概要:** インターネットを利用する際、読みやすさを向上させるために Web ページ上の特定のコンテンツのサイズを大きくしたり、背景色を変えたり、あるいは非表示にしたりしたいということがあるが、エンドユーザーがそのようなカスタマイズを行うのは容易ではない。本システムでは簡単な操作で Web ページのカスタマイズを可能にするインターフェイスと、各ユーザーが作成したカスタマイズリストを容易に公開・共有できるホスティング環境とを提供する。また作成されたカスタマイズリストを、ユーザー間だけでなくサイト運営者に対しても共有することで、ユーザーの操作を明示的なフィードバックとしてサイトの改善に役立つ等の新たな活用法も提案する。

## 1. 背景

Web ページを閲覧する環境は、表示するデバイスのサイズや利用者の視力によって様々である。そのためいかなる環境でも快適に閲覧できるように Web ページの表示内容を制御したいという欲求が存在する。例えば文字が小さく読みづらい場合フォントサイズを拡大し、行間を広げる、要素の境界やボタン等の色とページ背景色とのコントラスト比が小さく、識別しづらい場合は背景色を変更する、Web 広告やトリックバナーによって表示領域が圧迫されている場合はそれらを非表示にする等の操作を行うことができれば快適な閲覧を支援できる。このような操作を提供するべく様々なシステムが開発されているが、利用者が簡単に表示を変更できるシステムは存在しない。簡単な操作で表示を変更し、それを共有できるシステム **DOMinGO** を開発した。

## 2. DOMinGO の利用例

### 2.1 Web ページの表示制御

筆者が普段閲覧している Web ページをこのシステムによって実際にカスタマイズする。図 1 の上部左側が編集前の、上部右側が編集後の StackOverflow[11] のトップ画面である。このサイトはプログラミングや開発に対する QandA を掲載しているが、自分の場合投稿に対する投票数や閲覧数よりも投稿内容が自分の求めている情報と合致し

ているかどうかはすぐわかることの方が重要であるため、タイトル以外の Vote や Views を表示するコンポーネントを非表示にし、そのぶんリストビューの領域を拡大させた。また閲覧に関係の無い求人広告やサインインを求める要素も同様に非表示に設定した。その結果サイト全体の見通しが改善することができた。

図 1 の下部左側が編集前の、下部右側が編集後の Twitter[12] の画面である。

タイムラインの可読性を向上させるためツイートと直接関係の無いコンテンツは非表示に設定したうえで幅を拡大させ、同時にツイート要素に border を追加することで個々のツイートを識別しやすくした。また閲覧しているコンテンツが自分にとって本当に重要なかどうかを最小限のバイアスで判断するためにリツイート数やいいね数を表示するインジケータも非表示に設定した。その結果ツイートと関係の無い要素やリツイート数などのフィードバックの影響を受けることなく快適に閲覧することができた。

このことから、Web ページの表示を制御する手軽なインターフェイスは快適な閲覧環境を実現する上で重要であることがわかる。

### 2.2 カスタムスタイルの共有

生成したカスタムスタイルは Web 上にホスティングされ、他の利用者がインポートできるようになっている。詳細は設計と実装で述べるがホスティングに用いられるのは単なるデータベースではなく、Wiki システムである。これによって手軽に目的のカスタムスタイルを検索したり、さらには編集したりという応用が可能になる。

<sup>1</sup> 慶應義塾大学政策・メディア研究科

Keio University

<sup>2</sup> 慶應義塾大学環境情報学部

Keio University

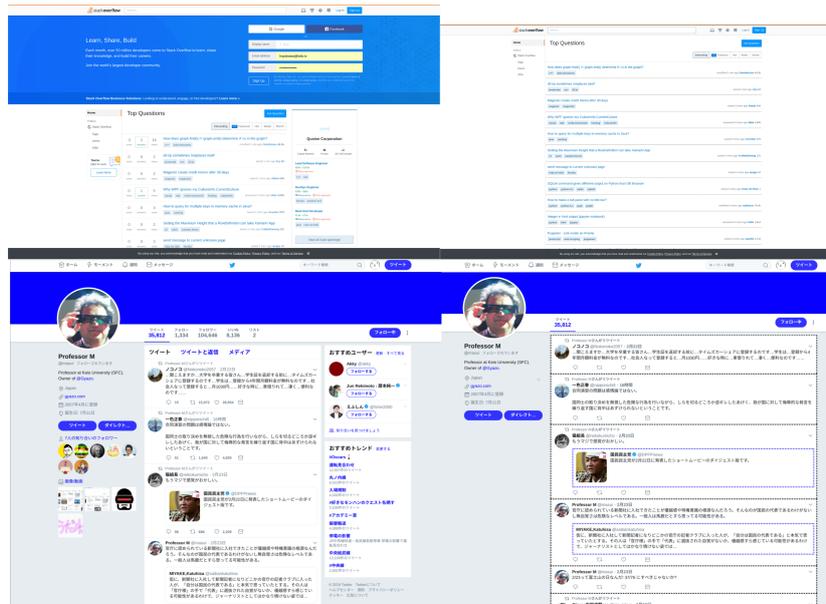


図 1 DOMinGO による操作適用前後の Web ページ

### 3. 設計と実装

簡単な操作で Web ページの表示を制御でき、またその操作結果として生成されるカスタムスタイルを手軽に共有できるシステムは、以下の要件を備える。

- (1) 要素を WYSIWYG/対話的に操作できること
- (2) 操作の結果を共有可能なカスタムスタイルとして出力できること
- (3) 他のカスタムスタイルをインポートできること

DOMinGO はこの要件を満たすべくブラウザ拡張機能と Wiki システムとを組み合わせられて開発されている。

#### 3.1 ブラウザ拡張機能：DOMinGO



図 2 DOMinGO の操作プロンプト

拡張機能のアイコンを押すと編集モードに移行し、マウスカーソルで選択した要素に対してプロンプトから望みの操作(アクション)を実行することができる。例えば文字が小さく密度が高い段落に対しては文字を大きくするや行間

を広げる等のアクションを適用することで可読性を高めることができる。また同じクラスを持つ要素が 2 回連続で選択された際に、他の兄弟要素に対する操作がサジェストされる。このような例示プログラミング [9] 機能によりリストビューの各要素に対する同時操作もサポートする。

#### 3.2 アクション

```
hide.json
{
  "action": "hide",
  "class": "domingo-hidden",
  "style": {
    "display": "none !important;"
  }
}
```

図 3 要素を非表示にするアクション

アクションは Web ページ中の要素に対する操作を記述したフォーマットである。JSON 形式に準拠しており、アクション名(操作の名称)と要素に適用するクラス名、スタイルシートが格納されている。DOMinGO の操作プロンプトはアクションの GUI ラッパーであるため、カスタムアクションを記述してインポートすればプロンプトから操作できるオプションを追加することができる。

#### 3.3 カスタムスタイル

実行されたアクションの結果は最終的に CSS 形式で出力される。他の利用者が出力したカスタムスタイルをインポートすることで自分の環境にもそのスタイルを適用する

```
custom-style.css
div.ProfileUserList.ProfileUserList--socialProof {
  display: none;
}

div.PhotoRail {
  display: none;
}

div.module.Trends.trends {
  display: none;
}

div.flex-module {
  display: none;
}

span.ProfileTweet-actionCountForPresentation {
  display: none;
}

div.Grid-cell.u-lg-size2of3 {
  font-weight: normal;
  border-style: none;
  width: 100%;
}

div.tweet-context.with-icn {
  font-weight: bold;
}
```

図 4 Twitter.com に適用するカスタムスタイル

ことができる。

### 3.4 Wiki システム : Scrapbox

Scrapbox[10] は Nota 社が運営する Wiki システムである。API を通じてテキストデータを読み込む/書き込む機能を提供するほか Javascript や CSS 等のソースコードを添付することも可能なため、カスタムアクションやカスタムスタイルをホスティングする環境として適している。またベースが Wiki システムであることから単純なデータの読み書きに留まらず、図 4 のようにコメントを書き加えたり、利用者間でスタイルに関する議論をしたり、スタイルを共同編集したりといったより応用的な使い方が可能である。

## 4. 考察と議論

### 4.1 システムに対する評価

筆者が所属している研究室のメンバーに、このシステムを用いて希望する Web ページの表示を制御するデモを行ったが、簡単な操作で Web ページをカスタマイズするというコンセプトについては多くの賛同が得られた。既存の Web ページに対して不満を持っている利用者は多いものの、カスタム CSS を自力で作る場合は相応のスキルが必要なため適切な解決手段を得られないという状況に置かれていた

ためである。

またカスタムスタイルを単なるデータベースではなく Wiki システムにホスティングするというコンセプトに対しては、より発展的な応用を促す効果があるという評価がなされた。Scrapbox は様々な用途に利用されているが、Scrapbox そのものをカスタマイズする Tips やスクリプト、CSS 等を共有するプロジェクト [14] が存在する。Wiki としても高度なコラボレーションツールとしても機能する Scrapbox をホスティング対象にする DOMinGO はこの点で他のカスタムスタイル共有システムに対してアドバンテージがあると考えられる。

簡単な操作で Web ページの表示を制御でき、またその操作結果として生成されるカスタムスタイルを手軽に共有できる DOMinGO の有用性や設計の妥当性を示すことができたと考えられる。

### 4.2 システムの課題

一方で解決すべき問題点もある。例えばある Web ページに適用可能なカスタムスタイルを公開・共有することは、自らがその Web ページを閲覧していることを大々的に公言しているに等しい。カスタムスタイルは利用者の閲覧履歴やプライバシーと密接に関わっているため慎重に取り扱わなければならない。カスタムスタイルを共有する利用者の心理的安全性を担保できなければ公開・共有活動を萎縮させかねないからである。

その対策の一つとして想定しているのが匿名カスタムスタイルである。通常のカスタムスタイルは適用する Web ページの URL を平文でメタデータに格納しているが、匿名カスタムスタイルの場合は元の URL を暗号的ハッシュ関数に基づいたメッセージダイジェストに変換した上で格納する。匿名カスタムスタイルは現在開いている Web ページの URL のメッセージダイジェストが一致した時のみ適用されるため、メタデータだけを見てもどの Web ページに対するものなのかを隠蔽することができる。

## 5. 関連研究

ChromeDevTools[3] 等のブラウザが標準搭載する Web 開発ツールを用いることで、ページ中の任意の要素に対して様々な操作を行うことができる。しかしこれらのツールは Web ページのデバッグを主眼に置いているため、Web ページのスタイルを手軽に変更するという用途には適さない。StyleURL[5] は Web ページを ChromeDevTools で操作した結果をカスタムスタイルとして出力し、Github[4] にアップロードする機能を提供するブラウザ拡張機能 [1] である。しかし利用にあたって ChromeDevTools や Github のような開発者向けツールを習得していることが前提であるため、一般的な利用者が気軽に扱えるものとは言いがたい。

Stylus[6] も Web ページに対してカスタムスタイルを適

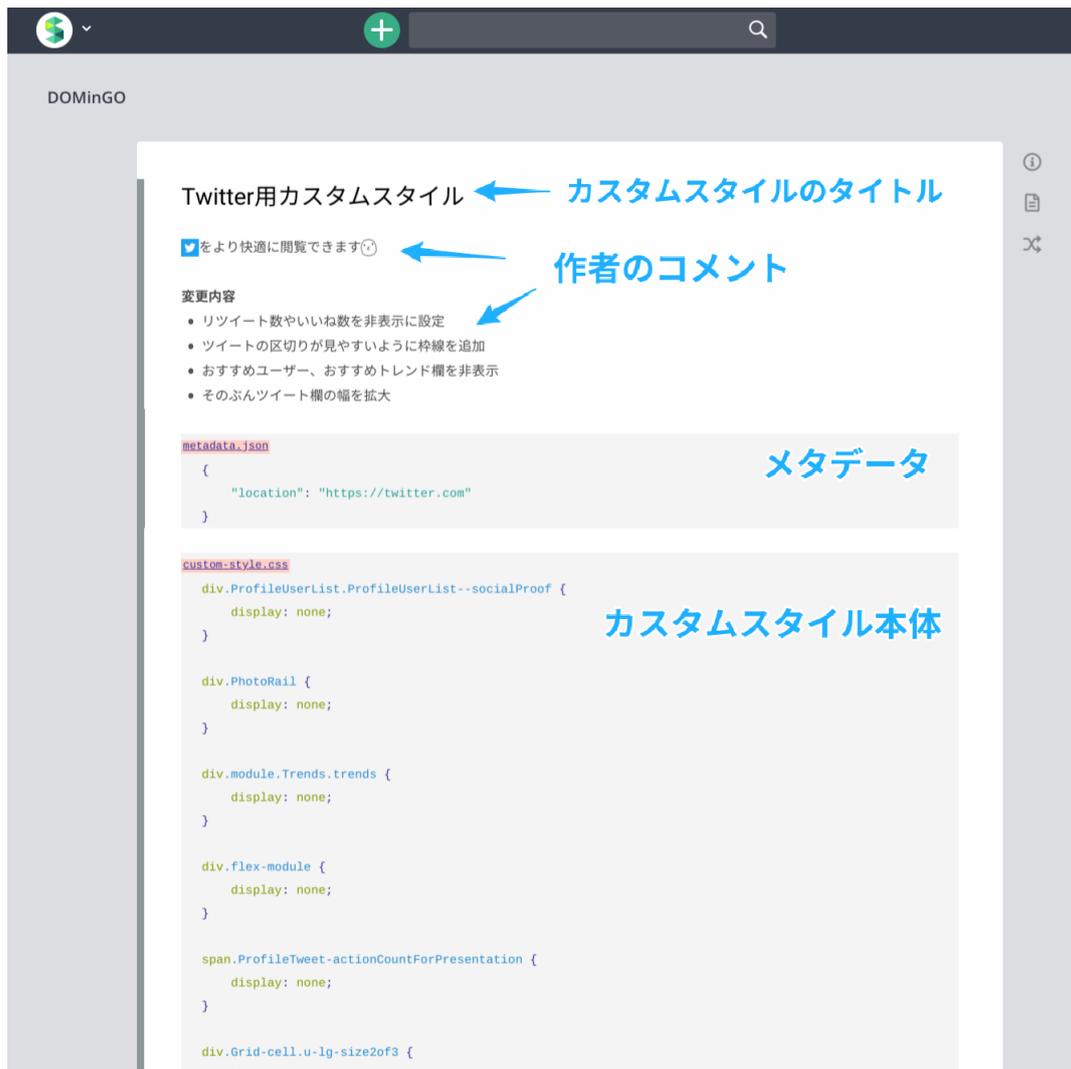


図 5 Scrapbox におけるカスタムスタイル表示画面

用するブラウザ拡張機能であるが、カスタムスタイルを独自にホスティングすることで利用者が手軽に検索・導入できる環境を実現している。一方でカスタムスタイルの作成には CSS そのものを編集できるスキルが求められるため、一般的な利用者にとって高いハードルが存在する。

StyleBot[7] は GUI ベースで現在表示している Web ページの表示を制御するインターフェイスを備えている。これにより CSS に熟達していない利用者が Web ページの表示を制御し、カスタムスタイルを作成することを可能にしているが、StyleBot はカスタムスタイルのホスティングを行っていないため、利用者間で共有することができない。Adblock Plus[8] は Web ページ閲覧の妨げとなる広告バナーやトリックバナーを非表示にするコンテンツブロッカーに分類されるブラウザ拡張機能である。要素が広告か否かをヒューリスティックに判断してブロックするほか、GUI によって利用者が任意の要素をブロック対象に追加することができる。カスタムスタイルならぬカスタムブロックリストを共有することは可能だが、そのホスティングは作成

者が行わなければならないため共有の敷居が高い。そのため殆どの利用者はホスティング環境を備える一握りのカスタムブロックリスト作成者のリストを受動的に使っているに留まっている。また Web 広告の排除を主眼に置いているため、一部の Web ページとの間で対立に発展している。具体的には Adblock Plus を導入しているブラウザのコンテンツ閲覧を制限する等の措置を取る運営者も存在する.[13].

ここで紹介したシステムはある程度普及しているものの、専門的な知識を必要とせず対話的な操作で Web ページの表示を制御し、またその操作によって生成されたカスタムスタイルを手軽に共有できる環境を兼ね備えるものは存在しない。

## 6. 結論

本稿では、背景において Web ページの表示を制御したいという欲求が一般的であることを示した。それを解決するシステム DOMinGO を開発し DOMinGO の利用例でその具体的な活用法を提示した。設計と実装において本シス

テムの基本構成について述べた。考察と議論では本システムに対する評価を元に有用性や課題を分析した。そして関連研究では

- (1) 要素を WYSIWYG/対話的に操作できること
  - (2) 操作の結果を共有可能なカスタムスタイルとして出力できること
  - (3) 他のカスタムスタイルをインポートできること
- 以上の3要件を満たしているシステムが現時点で存在していないことを分析した。

## 6.1 総括

本研究で開発した DOMinGO は Web ページを簡単な操作でカスタマイズしたいという欲求を解決し、さらに各々のカスタマイズを手軽に共有しあうという行為を支援することを可能にした。今後はシステムの課題で挙げた問題点を解決しつつ、機能の拡充に取り組んでいく。

## 参考文献

- [1] WhatAreExtensions?-GoogleChrome  
<https://developer.chrome.com/extensions>
- [2] OnlineGlossaryofDigitalMarketingTerms-MarketingTerms  
<https://www.marketingterms.com/dictionary/>
- [3] ChromeDevTools  
<https://developers.google.com/web/tools/chrome-devtools/>
- [4] GitHub  
<https://github.com>
- [5] StyleURL  
<https://www.styleurl.app/>
- [6] Stylus ChromeWebStore  
<https://chrome.google.com/webstore/detail/stylus/clngdbkpkpeebahjckkjfobafhncgmne?hl=ja>
- [7] Stylebot ChromeWebStore  
<https://chrome.google.com/webstore/detail/stylebot/oiaejidbmkiecgbjeifoejjpgmdaleoha>
- [8] Adblock Plus  
<https://adblockplus.org>
- [9] Toshiyuki M: *Repeat and predict - two keys to efficient text editing*, SIGCHI (1994)
- [10] Scrapbox  
<https://scrapbox.io>
- [11] Stackoverflow  
<https://stackoverflow.com/>
- [12] Twitter  
<https://twitter.com>
- [13] Umar I. and Zubair S. and Zhiyun Q.: *The ad wars: retrospective measurement and analysis of anti-adblock filter lists*, IMC (2017).
- [14] Scrapbox カスタマイズコレクション :  
<https://scrapbox.io/customize/>