CSSコーディングの複雑性軽減手法の提案

瀬下明紗子(ashe) 村井純研究室 kumo
ashe@sfc.wide.ac.jp

概要

巨大化、複雑化するWebアプリケーション開発の中で、CSS(Cascading Style Sheet)の担う役割もより重要なものになっている。しかし、CSS言語の特性により大規模な開発においてはCSSコードを構造化し保守・管理することは容易ではない。本研究は、CSSコーディングを複雑にしているCSSそのものの特徴に着目し、その解決策を考察するものである。

背景

近年、Webの普及や周辺技術の発展により、WebサイトやWebアプリケーションは巨大化また複雑化した。それに伴いWebドキュメントにおいて体裁を整える役割を担うCSSが使われる機会も増え、大規模なプロジェクトも増加している。

CSSはその特徴により、メンテナンス性を維持したコーディングをすることが困難である。 例えば、CSSのクラスやプロパティは値がグローバルに継承される。これにより要素の打ち消しが容易に発生し、冗長なコードや単に不要なコードの原因となる。 またCSSはプログラミング言語において一般的な変数や関数を持たず、コードの再利用が困難である。これもまた冗長なコードの原因となる。

複雑また巨大化したコードは開発効率を下げる他、ブラウザーでのアプリケーション表示効率を下げ、ユーザ体験を損ねる原因にもなる。

解決手法

- CSS設計方針の利用 (e.g. BEM, OOCSS, FLOCSS)
- コーディングガイド作成の習慣化
- プリプロセッサ(e.g. SASS, LESS) による擬似的機能拡張
- CSS Variables (W3C策定中)
- JavaScript によるクラスモジュールのローカル化 (e.g. CSS in JS, CSSモジュール)
- ShadowDOMの活用によるモジュールのローカル化

先行研究

Ali Mesbah, Shabnam Mirshokraie, "Automated analysis of CSS rules to support style maintenance", *Software Engineering (ICSE) 2012 34th International Conference on*, pp. 408-418, 2012, ISSN 1558-1225.

- CSS自動解析手法の提案。WebアプリケーションにおけるCSSとDOMの関連性を調べ、不使用や打ち消しで無効になっているCSSを検出している。また調査の結果、いくつかのオープンソースwebアプリケーションで平均60%ものCSSコードが実際には使われていないコードであることがわかった。

展望

方針・新規性などの確定は今後の課題

- CSSコードがスパゲッティだらけになる原因を減らしたい

- → CSSをスコープや変数や関数を持つ他のプログラミング言語みたいにして、大規模開発でも使いやすい言語にする

- CSS設計方針 (e.g. BEM, OOCSS, FLOCSS) → クラスのモジュール(ボタンとかフォームとか)わけの仕方や命名規則を定めた規約書のようなもの
- コーディングガイド作成の習慣化 → ここで言いたいコーディングガイドは、たとえばBootstrapにおけるモジュールリストみたいなもの(クラス名、サンプルHTML、簡単な用途の説明のリスト) コード全体の見通しをよくし、既存コードの再生産を防ぎます。
- プリプロセッサ(e.g. SASS, LESS) による擬似的機能拡張
- CSS Variables (W3C策定中)→ スコープを持った変数の実装
- JavaScript によるクラスモジュールのローカル化 (e.g. CSS in JS, CSSモジュール) → そもそもCSSをJavaScriptのほうで全て管理してしまおうという考え方です。モジュールごとのクラス名の管理をJSのほうで自動化し、各モジュール内のクラス同士が干渉しない擬似的なスコープを実現します。
- ShadowDOMの活用によるモジュールのローカル化 → ShadowDOM (任意のDOMを全体から切り離してカプセル化するHTMLの新仕様)で、HTMLやJSごとモジュールを分離してしまおうというやり方です。