変化に強いWebデザインとは
―改善や拡張が容易なWebサイトの設計―

Webデザインというと、「デザイン」という言葉を含むことから、とかく見た目ばかりが注目される。確かに、実際に姿や形を描くことはデザインにとって重要だが、どう描くかという計画（設計）にこそデザインの本質がある。本稿では、改善などの変更が容易なWebデザインとはどういうものか解説する。

NRIネットコム Webネット事業本部
Webテクノロジーコンサルティング部 フロントエンドエンジニア
高須賀（たかすか）　淳（じゅん）
専門はWebサイトの設計、フロントエンドの実装など

Webデザインを取り巻く環境

日本でインターネットの商用利用が始まって25年余りがたった。当初は表示技術の乏しさから文字や画像を組み合わせて表示する程度だったWebページは、映像も取り込み、ユーザーの操作に反応する双方向性を持ったメディアとして発展し続けてきた。

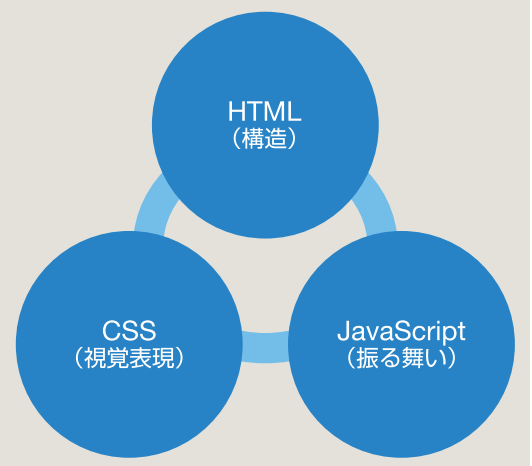
また、スマートフォンやタブレット端末などの新しいデバイスの登場、無線接続環境の整備などにより、さまざまな場所や状況でインターネットが利用できるようになった。インターネットで提供されるサービスも、通販だけでなく公共サービスや金融サービスなどさまざまな分野へと広がっている。

このようなインターネットの利用拡大にともなって、サービス利用のしやすさやWebサイト（Webページのまとまり）の使い勝手の良さなど、ユーザーエクスペリエンス（経験価値）の向上が求められるようになっている。

一方、Webデザインを実装するための技術に目を向けると、20年程前に比べてさほど大きな変化はない。HTML（HyperText Markup Language：Webページの構造を記述するための言語）やCSS（Cascading Style Sheets：Webページを構成する要素の色やサイズなど視覚的表現を定義するための言語）、JavaScript（Webページに動的要素などの機能を追加するための言語）は1996年にはすでにWebブラウザーに実装されていた。現在ではそれぞれの言語の役割が明確に定義され、国際的に標準化されている。（図1参照）

しかし、仕様が標準化されていても、意図

図1　Webデザインの実装における3要素



するWebサイトを実現するためにそれぞれの仕様をどう組み合わせるかは設計者の判断に委ねられる。それはWebサイトの規模や目的によっても違ってくるため、Webデザインの内容は千差万別である。以下で述べるのは、多様なWebサイトの中でも、基幹系システムと連動し、高いレベルの品質と可用性が必要で、利用者が多い大規模なWebサイトの設計についてである。

Webデザインにおける課題

まず、Webデザインにおける課題を挙げておこう。

①プログラミングによる視覚表現の難しさ

Webデザインは、主としてHTML、CSS、JavaScriptといったプログラム言語を組み合わせることによって実装される。すなわち、Webページの視覚表現がそれらの言語によってプログラミングされ、Webブラウザーがそのプログラムを解釈して描画処理を行いデザインされた視覚表現を行う。このように、Webデザインはプログラミングによって視覚表現を行うため、一般の造形のように結果を見ながら作業するのとは違う難しさがある。

②閲覧環境の変化を考慮した拡張性が必要

Webサイトはさまざまなデバイスで閲覧されるため、それらの違いに対応できる拡張性が求められる。例えば、処理能力が低いデバイスを考慮してコンテンツのデータ容量を抑えることや、新しいデバイスや閲覧環境による操作方法の変化に対応できるような設計とすることが必須である。

③表示パフォーマンスについての考慮が必要

ユーザーが利用している通信環境はさまざまであり、特にスマートフォンのようなモバイル機器は、場所によって通信速度が遅くなる場合や、通信が遮断される場合がある。新しい通信環境が高性能であるとは限らないのである。そのため、表示になるべく時間がかからないようにするなど、表示パフォーマンスについての考慮も必要となる。

以上の課題に加えて、Webデザインは見た目のことだと考えるシステム開発者が多く、Webデザインの設計についての正しい理解が進まないことも問題として挙げておきたい。

コンポーネント設計の有効性

HTMLにおいては、再利用性を高めるために、繰り返し使用される部品をコンポーネントとして定義しておくのがセオリーである。運用が長期間にわたる場合でも、各部品を定義し直すことで新たなニーズに対応できるため、全部を作り直さずにデザイン品質を維持することが容易である。

(1) コンポーネント設計とは

コンポーネント設計は玩具のレゴブロックに例えると理解しやすい。レゴブロックは、さまざまな部品を組み合わせて大きな形状を造形できるようにするために、ブロック同士を接続する部分の形状は変わらないように設計されている。

Webデザインにおけるコンポーネント設計も、見出しや文章、リスト（一覧）、入力フォーム、ボタンなどを再利用しやすい単位

図2　Webページ上のコンポーネントの例

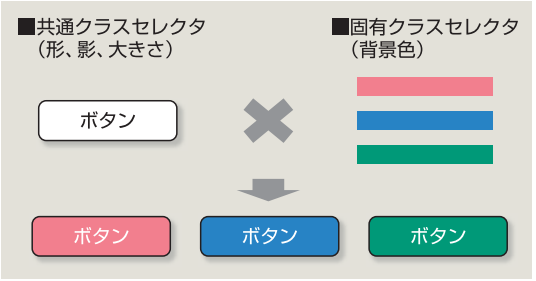
で切り出してHTMLソースコードの部品（コンポーネント）としてまとめ、これに装飾を適用するようにして、Webサイト内の複数の箇所で部品を繰り返し利用できるようにすることである（図2参照）。レゴブロックと同様に、部品の組み合わせに不都合がない合理的な設計が要求される。

(2) OOCSSを取り入れたコンポーネント設計

最近ではOOCSS（Object-Oriented CSS：オブジェクト指向CSS）を取り入れたコンポーネント設計が登場している。コンポーネントの粒度を変化させて、組み合わせの自由度を高めたものである。例えば、図3のように3つのボタンが配置され、ボタンの大きさや文字サイズは同じで、背景色がそれぞれ赤色、青色、緑色だとしよう。

この場合、ボタンの大きさや文字サイズは共通のクラスセレクター（CSSにおいてWebページの要素にスタイルを適用する方法の

図3　OOCSSによるコンポーネント設計の概念



1つ）としてまとめ、背景色部分のクラスセレクターをそれぞれ作る。CSSでは1つの要素に複数の属性を指定でき、赤いボタンには「共通クラスセレクター＋赤色背景クラスセレクター」、青いボタンには「共通クラスセレクター＋青色背景クラスセレクター」を適用する。このように、共通部分、異なる部分を別々に定義しておくことでクラスセレクターの再利用が可能になり、多様な表現の設計が容易になる。

(3) 一貫性のある視覚表現と拡張性の実現

前述したように、コンポーネント設計とは、よく使用される部品をそれぞれコンポーネントとしてまとめておくことにより、Webサイトの各所で繰り返し使用できるようにする設計方法のことである。例えば、ヘッダー部分に配置されるヘルプボタン、検索フォームの横に配置される検索実行ボタン、パーツの開閉を実行させるボタンなどは共通してよく使用される部品である。

コンポーネントを使い回すこつは、部品とレイアウトを別のコンポーネントとして用意することである。そうすれば、ボタンの装飾がレイアウトに影響することはなく、レイアウトが変わってもボタンの装飾が変わることはない。またOOCSSの概念を取り入れ、小さなボタン、それと組み合わせて使うフォー

ム類、それらを配置するレイアウト、さらにページ全体の構成を決めるページテンプレートなど、粒度の違うコンポーネントを組み合わせることで、表現の幅を広げながら視覚表現に一貫性を持たせることが容易になる。

このようなコンポーネント設計に加えて、各コンポーネントがどのように利用されているかを管理するパターンライブラリーを整備しておけば、修正時の対象範囲が判別しやすくなり、メンテナンスが容易になる。新しいデバイスへの対応が必要となった場合にも、コンポーネント単位で修正を行ってその組み合わせの妥当性を検証すればよく、対応のための大掛かりな作業を回避できる。このようにコンポーネント設計にはWebサイトの拡張性を高めるという利点もある。

(4) 表示パフォーマンスの向上にも効果

一般に、Webページの表示に掛かる時間はサーバーサイドで20%、ブラウザーで80%と言われている。ブラウザー表示では、画面レイアウトや装飾などの視覚表現に関わるファイルの読み込みやその描画が行われており、その中で、視覚表現を行う多くの部分はCSSが担っている。この部分でOOCSSを活用したコンポーネント設計を行うと、上述のようにクラスセレクターの再利用ができるため、ファイル容量の増大が抑えられて表示パフォーマンスが向上する。

実プロジェクトでの取り組み

ここで、野村総合研究所（NRI）が提供する証券総合バックオフィスシステム「THE STAR」のインターネット向け機能サイトのリニューアルにおけるWebデザインについて紹介しよう。当プロジェクトでは、HTML、CSS、JavaScript部分の制作はNRIネットコムが行い、アプリケーションへのHTML部分の取り込みは別の開発担当が実施した。

大規模プロジェクトの場合、要件変更やバグ修正などが多く発生する。これに対応するために、上記のように役割分担を厳密にするとともにOOCSSの概念を取り入れたコンポーネント設計を採用した。これにより、HTMLの部分に影響を及ぼさずにデザイン修正（CSSとJavaScriptの部分に限定した修正）を並行して行うことができ、その結果、開発期間の短縮や、HTMLによるアプリケーション開発の工数削減を実現できた。

また、大量のHTML文書を効率よく制作するために、コンポーネントをまとめたスタイルガイドを事前に作成した。スタイルガイドは単純なコンポーネント一覧ではなく、利用方法やコーディング（ソースコードの記述）例も記載したもので、標準化資料としてチーム内で共有された。

さらに、異なるブラウザーやデバイスでの稼働の確認をコンポーネント単位で実施して障害の発生を事前に防ぎ、生産性と品質を向上させることができた。

Webサイトは、新しいデバイスや機能の出現などをきっかけに常に変化する特性があり、またユーザーエクスペリエンスの向上に継続的に対応する必要がある。そのために、改善や拡張といった変化への対応が容易なWebデザイン（設計）が重要なのである。■