Engineer

# プログレッシブエンハンスメント

HTMLだけで使えるページを作り、後から機能を加える方法

## どのようなものか

プログレッシブエンハンスメントは、常に動く土台から始め、層を重ねてウェブページを作る方法です。土台のHTMLは、すべての内容を持ち、すべての用事を単独で済ませられるようにします。リンクには行き先があり、フォームはサーバーへ送信され、ほかを何も読み込まなくても、ページを読んで使えます。その上に、配置を整えて見やすくするスタイルシートを加えます。スクリプトは最後に置き、操作を速く、滑らかにします。各層は下の層を改善しますが、用事を済ませるために、HTMLより上の層を必要とはしません。GOV.UKのService Manualは、まずHTMLでページを動かす考え、とまとめています。

名前は、Steven ChampeonとNick Finckの講演「プログレッシブエンハンスメントによる未来のための包摂的ウェブデザイン」に由来します。二人の論文によると、2003年3月11日に、テキサス州AustinのSXSWで行われました。論文ではProgressive Enhancementと大文字で始め、PEと略しています。対比するのは、最も高機能なブラウザーを基準に作り、それ以外でも内容に届くことを期待する、従来のグレースフルデグラデーションです。論文のPEは、最も機能の少ない機器でも使える文書から始め、見せ方を別のロジックとして加えます。基本的な機器に、最新ブラウザー向けの処理まで背負わせないためです。

違いは、作業の向きです。グレースフルデグラデーションは、完成した体験から始め、一部が欠けたとき何が残るかを考えます。プログレッシブエンハンスメントは、必ずあるものから始め、何を加えられるかを考えます。良い回線上の現在のブラウザーでは、両方が同じページに行き着くこともあります。違いが出るのは、何かが失敗したときです。土台から積み上げたページには土台が残りますが、上から作ったページに残るのは、作者が用意し忘れなかったものだけです。

ライブラリーや機能ではなく、作り方です。HTML、CSS、JavaScriptに、その名前の機能があるわけではありませんが、標準は実現できる余地を残しています。ブラウザーは、理解できないCSS宣言を読み飛ばし、認識できない種類のスクリ

プトを無視します。スタイルシートは `@supports` 規則で、使う前に、その機能があるかを確認できます。

概念を判断に生かす

「プログレッシブエンハンスメントとは？」を実際の判断に生かしましょう。その概念を、開発または購入するシステムに結び付けて考えるお手伝いをします。 読み込みが速く、スマートフォンでも使え、お客様がどのブラウザーを使っていても同じように動く画面を提供します。

**サービスを見る** お問い合わせ →

用語の背後にある前提を検証する

解説とリンク先の実績を参考に、プロジェクトについての質問を整理してください。ツールを選ぶ前に、前提を一緒に検討できます。 フルスタックのGIS地図開発を指揮 — DBからブラウザーまでのフルスタック地図開発

**フルスタックのGIS地図開発を指揮** お問い合わせ →

## なぜ重要なのか

層ごとに失敗の仕方が違い、GOV.UKのService Manualは、それを具体的に説明します。CSSの層は比較的失敗に強く、ブラウザーは知らない宣言を飛ばして続けます。ただし、知らないセレクターがあれば、その規則全体を捨てます。JavaScriptの層は、そうはいきません。ブラウザーにない構文やインターフェースをスクリプトが使うと、その先は実行されません。主要な用事をスクリプトに依存するページの信頼性は、サーバーから読み手までの、あらゆる段階の信頼性に左右されます。

その段階はいくつもあります。同じページは、CSSやJavaScriptの読み込み・実行を止める原因として、一時的なネットワーク障害、広告ブロッカー、停止したコンテンツ配信網、DNS障害、ブラウザー更新の不具合、組織や個人のファイアウォールを挙げます。どれも古いブラウザーとは関係がなく、読み手には、どれが原因か分かりません。見えるのは、何もしないボタンだけです。マニュアルは発生頻度の数字を示しておらず、主張の根拠は、ある割合ではなく、原因の一覧です。

意図して機能を無効にする人もいます。マニュアルは、ブラウザーの機能を自分で無効にする利用者がおり、その選択を尊重すべきだと述べます。HTML標準も、多くの利用者がスクリプトを無効にすることと、宣言的な機能の方が保守しやすいことが多い点から、可能ならスクリプトの代わりに宣言的な機能を使うよう勧めます。例は `details` 要素で、JavaScriptを一行も書かず、ページの一部を開閉できます。標準は、スクリプトを使えない場合も、アプリケーションの基本機能が失われないようにすることを求めています。

英国政府のサービスについては、マニュアルは規則として示しています。HTMLだけで機能し、JavaScriptを使うならHTMLとCSSの改善だけに使い、スクリプトが失敗してもサービスを使えるようにする、というものです。ある政府のサービスに向けた規則で、ウェブ全体を対象とするものではありません。ただし、誰が読むかを作者が選べないサイトであれば、その理由は共通します。

チーム側にも利点があります。追加した改善が壊れたとき、利用者が失うのは改善部分であって、用事を済ませる手段ではありません。そのため、スクリプトの不具合は、利用停止ではなく、不便にとどまります。新機能に対応するブラウザーへ配信する際、非対応のブラウザー向けに別の計画を立てる必要もありません。そちらには、今まで使えていたページが残るからです。

## 実際の活用例

HTML標準自身が示す、スクリプトなしでも基本機能を保つ例は、列見出しで並べ替える表です。この方法を確かめるのに適しています。土台となる表の見出しは、普通のリンクです。クリックすると、同じ表をその列で並べ替えるようサーバーへ求め、サーバーは、その順の表を返します。スクリプトの有無にかかわらず、どのブラウザーでも動きます。

```
<table id="orders">
  <thead>
    <tr>
      <th><a href="?sort=date">日付</a></th>
      <th><a href="?sort=total">合計</a></th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>2026-10-20</td>
      <td>120</td>
    </tr>
    <tr>
      <td>2026-10-24</td>
      <td>45</td>
    </tr>
  </tbody>
</table>
<script type="module" src="sort.mjs"></script>
```

改善する層がスクリプトです。同じリンクを見つけ、画面遷移を取り消して、ページ内で行を並べ替えます。サーバーとの往復を一回省けます。

```
const table = document.querySelector("#orders");
const body = table.tBodies[0];
const collator = new Intl.Collator("en", { numeric: true });

function sortBy(column) {
  const text = (row) => row.cells[column].textContent;
  const rows = [...body.rows];
  rows.sort((a, b) => collator.compare(text(a), text(b)));
  body.append(...rows);
}

for (const link of table.tHead.querySelectorAll("a")) {
  link.addEventListener("click", (event) => {
    event.preventDefault();
    sortBy(link.closest("th").cellIndex);
  });
}
```

リスナーを付ける前に起きた失敗なら、そのリンクは使えるままです。`sort.mjs` が届かなければ、リンクは従来どおりに動きます。スクリプトが途中で例外を投げても、まだリスナーを付けていない見出しのリンクは、サーバーでの処理に戻れます。モジュールスクリプトに対応しないブラウザーは、`type="module"` を未知のスクリプト種別として扱い、HTML標準にあるとおり無視します。`async` のないモジュールスクリプトは、ページの解析が終わってから動くため、表を探す時には、すでに表があります。ただし、この例は、取り付け済みのクリックハンドラーが、`preventDefault()` で画面遷移を取り消した後に例外を投げた場合、回復しません。

負担はサーバー側にあります。土台には並べ替えを行うエンドポイントが必要で、二つの並べ替え方法は、同じ結果を返す必要があります。これが、この方法で一度支払う通常のコストです。目指す利点は、サーバーへのフォールバックで主要な用事を済ませられるページです。取り付けたハンドラーは、自身の処理が失敗したときも、その経路を保つ必要があります。

このサイトは、土台を明示しています。アクセシビリティー方針には、実行可能なJavaScriptを含まない、HTML、CSS、SVGの完全なページを提供すると記されています。検索コードは残していますが無効で、ページ内の絞り込み機能に依存しません。方針は、残る制限も挙げます。アイコンへポインターを重ねたときやフォーカスしたときの小さなラベルは、キーで閉じられません。それには、このサイトが配信しない、ブラウザーで動くコードが必要です。そのため、1.4.13「ホバー又はフォーカスで表示されるコンテンツ」は、一部のみ満たします。

# 問題が起きるところ

**`noscript` ブロックを、土台の代わりにします。** `noscript` 要素は、スクリプトが無効なときだけ内容を表示します。HTML標準は「noscript要素は大ざっぱな手段である」と率直に述べます。スクリプトが有効でも、ページのスクリプト自体は失敗することがあり、その場合、読み手はスクリプト版のページも代替手段も得られません。まずスクリプトなしで動くページを作り、その後でスクリプトに変更させます。標準自身の例も、スクリプトでフォームへ動的に更新される結果を加え、送信ボタンを取り除きます。

**未対応の機能が一つあると、スクリプト全体が止まります。** JavaScriptの層は失敗を許容しないため、バンドルに一つ新しい構文が入っただけで、本来は動くものも含め、そのバンドルの改善をすべて止めることがあります。土台はスクリプトから独立させ、失うのは改善部分だけにします。新しいコードには、古いブラウザーが読み飛ばす `type="module"` を使います。古いブラウザーにも改善版が必要なら、 `nomodule` を付けた別の従来型バンドルを与えます。モジュール対応ブラウザーは、それを実行しません。

**新しいセレクターで、使える規則まで失われます。** ブラウザーは知らないセレクターに出会うと、宣言も含め、その規則全体を捨てます。すべてのブラウザーが理解するセレクターと、新しい擬似クラスを並べて書くと、本来はスタイルを適用できたブラウザーからもスタイルが消えることがあります。一部のブラウザーが対応しない可能性のあるセレクターは、独立した規則にします。

**代替スタイルが、改善後の配置にも混ざります。** 新しい配置方式に非対応のブラウザー向けに書いた幅が、対応ブラウザーで、その配置方式と衝突することがあります。CSS Conditional Rules Level 3は、flexの配置と、代替用の `@supports not (display: flex)` ブロックで解決する例を示します。幅の宣言がflexの配置を壊しかねないため、非対応ブラウザー向けのスタイルだけに入れるべきだと説明しています。

**画像をスクリプトで選びます。** HTML標準は、マークアップで画像を指定せずスクリプトに選ばせると、スクリプトを無効にした読み手には、画像が一つもダウンロードされないと指摘します。また、マークアップですでに指定した画像をスクリプトで置き換えると、ブラウザーが二重にダウンロードを始めることがあります。HTMLの `srcset` と `sizes` 、または `picture` で候補を宣言し、ブラウザーに選ばせます。

**動作の下に土台がありません。** ChampeonとFinckの戦略一覧は、あらゆる動作に代替手段を求めます。クリックハンドラーには本物の `href` 、ブラウザー側の検査にはサーバー側の検査が必要です。クリックハンドラーだけで存在する操作部品や、ブラウザー内でしか検証しないフォームには、土台がありません。まず実際のリンクとフォーム送信を使い、サーバーでは常に検証し、ブラウザー側の検証は、その上に加える便利な機能として扱います。

**古いブラウザーだけの問題と考えます。** 土台は古いブラウザーのためだと考えるチームは、そのブラウザーがなくなれば土台も外します。GOV.UKが列挙する原因は、ネットワーク、ブロッカー、ファイアウォール、壊れた更新で、現在のブラウザーにも起きます。土台の経路は、実際に使い、試すものとして残します。ときどきスクリプトをブロックしてページを読み込み、主要な用事を最初から最後まで行います。

## 参考資料

- プログレッシブエンハンスメントの利用 — 2024年9月27日が最終更新日の、GOV.UKのService Manualの案内です。三つの層、それぞれの失敗、政府のサービスがHTMLだけで動くべき理由を説明します。
- HTML標準：スクリプト — `script` 要素の `type="module"` と `nomodule` 、および依存しないよう助言する `noscript` 要素です。
- CSS Conditional Rules Module Level 3 — `@supports` と、その `not` 、 `and` 、 `or` を定義する、2024年8月15日のW3C勧告候補草案です。
- HTML標準：画像 — `srcset` 、 `sizes` 、 `picture` と、スクリプトで画像を選ぶと、スクリプトを無効にした読み手へ届かなくなる理由です。

用語集

実際の制約に合わせて選ぶ

業務の流れ、既存のインターフェース、変更を必要とする利用者をお知らせください。検証可能な納品内容と、お客様のチームが保守を続けられる引き継ぎ内容を定められます。

**料金を見る** お問い合わせ →

# 用語を実務に結び付ける

「プログレッシブエンハンスメントとは？」がお客様の組織にどう当てはまるかをご相談ください。現在のシステムの具体例が一つあれば、話を始められます。

サービスを見る

welcome@engineer.company +45 50 10 61 84 デンマーク、コペンハーゲン

このページを持ち歩く

連絡先カード

心を込めて丁寧に制作： D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

著作権 © 2025 – 現在 · Engineer ApS · すべての権利を留保します。

印刷元： https://engineer.company/ja/glossary/progressive-enhancement/