

# HTMLオブジェクト

HTMLオブジェクトは、HTMLのタグを直接ドットプログラム内に記述し、Webページの要素としてそのまま扱うことができるオブジェクトである。これにより、動的な画面更新や双方向コンテンツの開発が容易になる。

HTMLオブジェクトは、角括弧 (<>) で囲まれたタグを直接記述することで生成する。タグは開始タグ (<>) と終了タグ (</>) をセットで使用する。次のプログラムを実行すると、画面に段落要素 `p` として「テキスト」という文字が表示される。

```
段落<p>テキスト</p>
```

HTMLオブジェクトの基本的な操作には、内容の変更、プロパティの設定、イベント処理の定義、階層構造の構築などがある。これらを組み合わせることで、多様なWebアプリケーションを開発できる。

HTMLオブジェクトの内容を変更するには「テキスト」メソッドを使う。次のプログラムを実行すると、段落に「こんにちは」という内容が設定される。

```
段落<p></p>
段落! "こんにちは" テキスト。
```

HTML要素のプロパティに直接アクセスすることもできる。たとえば、入力欄の値を取得するには `value` プロパティ、HTML内容を設定するには `innerHTML` プロパティ、スタイルを設定するには `style` プロパティを使う。

```
入力欄<input>
入力値 = 入力欄.value

コンテナ<div></div>
コンテナ.innerHTML "<span>内容</span>"

要素<div></div>
要素.style.display "none"
```

HTMLオブジェクトの中に別のHTMLオブジェクトを入れるには「いれる」 `append` `appendTo` などのメソッドを使う。「いれる」と `append` は追加先のオブジェクトに対して呼び出し、 `appendTo` は追加されるオブジェクトに対して呼び出す。

```
リスト<ul></ul>
項目<li>りんご</li>
リスト! (項目) いれる。
// または
項目! (リスト.appendTo)
```

HTML要素を削除するには `remove` メソッドを使う。これにより、動的にコンテンツを制御できる。

```
要素<div>削除される要素</div>
要素.remove
```

イベント処理には「動作」メソッドを定義する。ボタンがクリックされたときや、スライダーが操作されたときなど、ユーザーの操作に応じた処理を定義できる。

```
ボタン<button>送信</button>
ボタン：動作 = 「
ラベル！"ボタンがクリックされました" 作る。

```

```
スライダー<input type="range">
スライダー：動作 = 「
ラベル！（スライダーvalueテキスト。

```

HTMLオブジェクトは条件分岐と組み合わせて使うことで、状態に応じて異なる表示や機能を提供することができる。例えば、特定の条件が満たされた場合にのみボタンを有効にしたり、期間が過ぎたら操作できないようにしたりといった制御が可能である。

```
「条件」！なら「
ボタン<button>実行</button>
」そうでなければ「
<span>実行できません</span>
」実行。
```

HTMLオブジェクトにスタイルを適用するには、スタイルタグを使用する方法と、要素のスタイルプロパティを直接操作する方法がある。

```
スタイル<style>
.box { width: 100px; height: 100px; background-color: blue; }
</style>
```

```
ボックス<div class="box"></div>
```

```
// または直接スタイルを設定
要素<div></div>
要素style"width: 100px; height: 100px; background-color: blue;"
```

HTMLのID属性を使って要素を取得するには「#」を使用する。これにより、ページ内の特定の要素に簡単にアクセスできる。

```
<div id="output"></div>
出力#output
出力：テキスト"ID指定で取得した要素に内容を設定"。
```

画面の表示/非表示を切り替えるにはstyle:displayプロパティを使用する。これにより、単一ページでも複数の画面を切り替える疑似的なページ遷移を実現できる。ログイン画面、予約フォーム、商品詳細画面など、複数の画面を持つアプリケーションの開発に役立つ。

```
ページ1<div>
<h1>ページ1</h1>
<button id="toPage2">ページ2へ</button>
</div>
```

```
ページ2<div style="display: none;">
<h1>ページ2</h1>
<button id="toPage1">ページ1へ</button>
```

```
</div>□  
  
#toPage2□動作 = 「  
ページ1□style□display□"none"□  
ページ2□style□display□"block"□  
□□  
  
#toPage1□動作 = 「  
ページ2□style□display□"none"□  
ページ1□style□display□"block"□  
□□
```

モーダルやポップアップウィンドウは、絶対位置指定とCSSの「transform」を使って実装できる。これはアラート表示や確認ダイアログなどのインターフェースに利用できる。

```
<style>  
.popup-container {  
  position: absolute;  
  top: 50%;  
  left: 50%;  
  border: 1px solid;  
  transform: translate(-50%, -50%);  
}  
</style>□
```

```
ボタン□<button>ポップアップ</button>□  
ボタン：動作 = 「  
ポップアップ□<div class="popup-container"></div>□  
タイトル□<h3>ポップアップウィンドウ</h3>□  
閉じるボタン□<button>閉じる</button>□  
閉じるボタン：動作 = 「  
  ポップアップ□remove□  
  □□  
  ポップアップ！（タイトル）（閉じるボタン□append□  
  □□
```

フォーム要素には様々な種類があり、テキスト入力、日付入力、チェックボックス、スライダーなどを使ってユーザーからの入力を受け付けることができる。これらを活用することで、予約システム、アンケートフォーム、設定画面などを作成できる。

```
テキスト入力□<input>□  
  
日付入力□<input type="date">□  
  
チェックボックス□<input type="checkbox">□  
チェックボックス：動作 = 「  
結果！（チェックボックス□checked□テキスト。  
□□  
  
スライダー□□<input type="range" list="数値">。  
<datalist id="数値">  
  <option value="0"></option>
```

```
<option value="50"></option>
<option value="100"></option>
</datalist>
スライダー：動作 = 「
結果！（スライダ value テキスト。

```

外部コンテンツを表示するには `iframe` を使うことができる。これにより、他のWebページをドリトルの画面内に埋め込むことができる。

```
<style>
  iframe { width: 100%; height: 500px; }
</style>
```

```
フレーム <iframe></iframe>
フレーム src "https://example.com/"
```

ハイパーリンクを作成するには `a` タグを使う。これにより、他のページへの移動を実現できる。

```
リンク <a href="https://example.com/">リンクテキスト</a>
```

Webアプリケーション開発では、単一のHTML要素ではなく、複数の要素を組み合わせたコンポーネントを作成することが多い。例えば、商品カードや投票ボタンなどのコンポーネントを関数として定義し、必要に応じて再利用することができる。

```
商品カード = 「 | 名前 価格 画像URL
カード <div class="card"></div>
画像 <img class="card-image">
画像 src 画像URL
タイトル <h3 class="card-title"></h3>
タイトル：テキスト = 名前。
価格表示 <p class="card-price"></p>
価格表示：テキスト = "¥" + 価格。
カード！（画像）（タイトル）（価格表示 append
カード。

```

```
商品一覧 <div class="product-list"></div>
商品一覧！（商品カード！"りんご" 100 "apple.jpg" 実行 append
商品一覧！（商品カード！"みかん" 150 "orange.jpg" 実行 append
```

HTMLオブジェクトは、Webプログラミングの考え方や技術を学ぶための優れた教材である。HTMLとCSSの基本を理解し、ドリトルのプログラミング機能と組み合わせることで、ショッピングサイト、予約システム、投票アプリケーション、チャットシステムなど、多様なWebアプリケーションを容易に開発できる。また、サーバとの通信 `WebAPI` を組み合わせることで、より実践的なアプリケーションを構築することができる。

From:

<https://dolittle.eplang.jp/> - プログラミング言語「ドリトル」

Permanent link:

[https://dolittle.eplang.jp/ch\\_html\\_js?rev=1741331924](https://dolittle.eplang.jp/ch_html_js?rev=1741331924)

Last update: **2025/03/07 16:18**

