

Browser and UI

#2 HTML/ARIA


ken7253


Frontend developer


ブラウザの標準化まわりを追うのが趣味

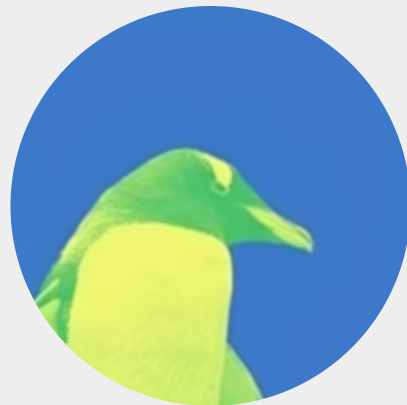
最近はReactを使ったアプリケーションを書いています。

ユーザーインターフェイスやブラウザが好き。

 <https://github.com/ken7253>

 <https://zenn.dev/ken7253>

 <https://bsky.app/profile/ken7253.bsky.social>



Browser and UI とはなにか

狭義の フロントエンドエンジニアの勉強会

狭義の フロントエンドエンジニアの勉強会

話していきたいこと

- ブラウザの仕様・標準化の話
- ブラウザの実装について
- UI・デザインの話
- UI実装を支えるツール
- フォント・画像とかのアセット系の話とかも

狭義の フロントエンドエンジニアの勉強会

積極的にはやらないこと

- ライブラリ・フレームワーク論
- （サービス全体の）設計論・アーキテクチャ
- サーバーサイドの話
- 技術以外の話

お願いしたいこと

お願いしたいこと

- 誰かを不快にさせる行動・発表はしないでください。
- ミニマムな開催にご協力をお願いします。
- 次回以降の会場提供できそうな人は教えて下さい。
- また次回やるので来てください！

Opening Talk

HTML Parser Quiz!!

HTML Parser Quiz!!

- 賞品はありません 🙄
- 基本的には頭の中で考えるだけで 🧠
- 余裕があったらハッシュタグでポスト 🧠
- 簡潔にするために解説では手順をスキップしている箇所があります 🙄

HTML Parser Quiz!!

手元で試してみたい人はコンソールで `DOMParser` を使うと簡単に試せるかも。

```
const p = new DOMParser();
const dom = p.parseFromString(`
<!DOCTYPE html>
<html>
<head></head>
<body><h1 id="foo">Hello HTML!</h1></body>
</html>
`, 'text/html');
console.log(dom.getElementById("foo"));
// => <h1 id="foo">Hello HTML!</h1>
```

<https://developer.mozilla.org/ja/docs/Web/API/DOMParser>

Q0: Example

```
<div data-foo="foo" data-foo="bar"></div>
```

例題：同一の属性値を複数記述した場合

A0: Example

```
<div data-foo="foo"></div>
```

最初に定義した属性値が採用される。（重複した属性値は無視される）

Q1: Skip Element

Q1: Skip Element

```
<!DOCTYPE html>  
<meta charset="utf-8">  
<div>Hello World.</div>
```

point

- `<html>` / `<head>` / `<body>` などを書いていない

A1: Skip Element

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <div>Hello World.</div>
  </body>
</html>
```

- `<html>` / `<head>` / `<body>` が挿入される
 - 特定の要素については書かれていなくても補完される
- ※ `<div>` の後に `<title>` などを入れた場合は `<body>` 内部に入る

A1: Skip Element

head内部の処理

- `doctype` が終了(`before html` 状態)のときに `html` 以外の開始タグが出現
 - `<html>` をDOMに挿入する
 - `before head` に移行する
- `before head` の状態で `head` 以外のタグが出現した
 - `head` をDOMに挿入する
 - `in head` に移行する
- `meta[charset="UTF-8"]` を挿入する

Q2: HTML Comment with inline script

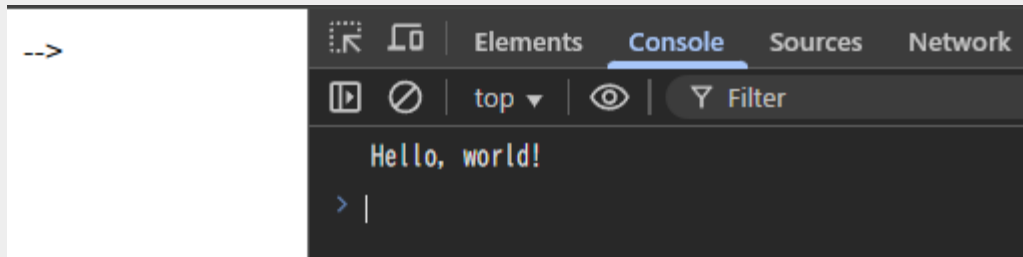
Q2: HTML Comment with inline script

```
1  <!doctype html>
2  <html lang="ja">
3    <head>
4      <script>
5        <!-- console.log("html comment");
6          console.log("Hello, world!");
7      </script>
8      -->
9    </head>
10   <body></body>
11 </html>
```

point

- `<!-- console.log("html comment");` はどのように解釈されるか
- `-->` はどのように解釈されるか

A2: HTML Comment with inline script



- `<script>` 内部はすべてテキストトークンとして扱われる。
- `<script>` 内部のテキストはすべてJSのParserに渡される。
- HTMLコメントは単行のコメントなので最初のログは出力されない。
- `console.log("Hello, world!");` は実行される。
- `in head` 状態の場合に開始タグ以外を発見したので `in body` に移行する。
- `-->` はテキストとして解釈され画面に表示される。

Q3: Formatting Elements

Q3: Formatting Elements

```
<!-- // -->
<body>
  <p>foo<b>bar<a href="#top">buzz</b>qux</a>foobar</p>
</body>
<!-- // -->
```

- `<a>` を開いたが先に `` の閉じタグが来ている
- `` の閉じタグの後に `<a>` が閉じられている
- 閉じタグの順番がぐちゃぐちゃ
- HTML LSにも乗っている有名(?)な問題

A3: Formatting Elements

```
1  <!-- // -->
2  <body>
3    <p>foo
4      <b>bar
5        <a href="#top">buzz</a>
6      </b>
7      <a href="#top">qux</a>
8      foobar
9    </p>
10 </body>
11 <!-- // -->
```

- `qux` の部分に `<a>` の開始タグが挿入される
- これはFormatting Elements特有の挙動

Formatting Elements

The following HTML elements are those that end up in the list of active formatting elements: a, b, big, code, em, font, i, nobr, s, small, strike, strong, tt, and u.

HTML LSに定義されている下記の要素

a / b / big / code / em / font / i / nobr / s / small / strike / strong / tt / u

これらの要素が出現している間は自動閉じタグ挿入の仕組みが特別なものになる。

自動的に閉じタグが挿入されるが、本来の閉じタグがある場所まではフォーマットを維持するために開始タグを挿入したりする。

Step

```
<!-- // -->
<body>
  <p>foo
    <b>bar
      <a href="#top">
        buzz
      </a>
    </b><a href="#top">qux
  </a><!-- [] -->foobar
</p>
</body>
<!-- // -->
```

END