Technologie informacyjne
przed 9 wykładem

Andrzej Giniewicz

24.04.2024

Przechodzimy do tematyki tworzenia stron internetowych. Zanim jednak zaczniemy, zajmiemy się omówieniem krótko historii Internetu oraz ogólnej budowy strony. Wciąż na tym wykładzie poznamy też język HTML do opisu treści na stronach internetowych. Kolejne zajęcia będą dotyczyły języka CSS do opisu stylu stron, ECMAScript do opisu interakcji na stronie oraz biblioteki Flask z Pythona, do programowania logiki po stronie serwera.

1 Historia Internetu i Wojny Przeglądarek

Pomysł na Internet pojawił się na początku lat sześćdziesiątych XX wieku. Był to okres narastającego napięcia nuklearnego pomiędzy Stanami Zjednoczonymi a Związkiem Radzieckim, w którym z roku na rok produkowano tysiące głowic jądrowych[1]. Sieć, która wtedy powstawała, miała nazwę „Sieci Intergalaktycznej"[2]. Ambitny pomysł zaczęto realizować między innymi z powodu właśnie napięć okresu Zimnej Wojny. Około 1969 roku uruchomiono pierwsze węzły sieci ARPANET, zaprojektowanej tak, aby przetrwać atak nuklearny. Ważnym założeniem w związku z tym wymogiem było, że sieć nie może mieć jednego centralnego punktu, w który uderzając, można by było unieszkodliwić całą sieć.

Po latach do sieci dopuszczono instytucje edukacyjne, które do tego czasu tworzyły swoje sieci, łączące uniwersytety. W latach osiemdziesiątych pieczę nad siecią przejęła agencja NSF (National Science Foundation). Pierwsze firmy sprzedające dostęp do Internetu pojawiły się w Stanach Zjednoczonych i Australii w 1989 roku, jednak do 1995 roku, Internet wciąż był przeznaczony jedynie na użytek niekomercyjny. Komercyjny Internet, jaki znamy dzisiaj, pojawił się dopiero w 1995 roku, czyli 27 lat temu.

W roku 1991 powstaje pierwsza wersja języka HTML, pierwsza wersja CSS w 1994 roku a pierwsza wersja JavaScript w 1995 roku. Pierwsza graficzna przeglądarka została wyprodukowana w 1993 roku i nazywała się Mosaic, rok później pojawiła się przeglądarka Netscape Navigator (to w ramach pracy nad tą przeglądarką powstało wiele ważnych technologii w tym JavaScript). Oznacza to, że 1995 roku, w momencie otwarcia Internetu na potrzeby komercyjne, najważniejsze obecnie technologie były już na miejscu.

[1] Źródło: `https://ourworldindata.org/nuclear-weapons#stockpiles-of-nuclear-weapons`.

[2] Źródło: `http://www.historyofcomputercommunications.info/Book/2/2.1-IntergalacticNetwork_1962-1964.html`.

W okresie tym wiele firm walczyło o to, by mieć jak największy wycinek rynku przeglądarek. Były one wtedy osobnymi, płatnymi aplikacjami i każdemu zależało na sprzedaniu jak największej liczby egzemplarzy. Cena przeglądarki w połowie lat dziewięćdziesiątych wynosiła około 50 dolarów amerykańskich, co z otwarciem sieci na rozwiązania komercyjne, oznaczało potencjalnie ogromne zyski. W ramach wejścia na rynek internetowy w roku 1995 firma Microsoft kupiła przeglądarkę Mosaic i zaczęła ją sprzedawać pod nazwą Internet Explorer. Moment ten powszechnie uznaje się za początek tak zwanej pierwszej wojny przeglądarek. Przegrywający walkę Microsoft, posiadając jedynie 10% rynku, w 1997 roku postanowił dodać przeglądarkę do systemu operacyjnego Windows i sprzedawać ją w pakiecie wraz z nim. Ruch ten spowodował odwrócenie sytuacji do 2002 roku, kiedy Internet Explorer 6.0 posiadał 90% rynku przeglądarek, oficjalnie wygrywając pierwszą wojnę przeglądarek. Okres ten niósł za sobą poważne konsekwencje. W tym czasie producenci prześcigali się, dodając do języka coraz to nowsze funkcjonalności. Wszystkie „potworki" takie, jak migający tekst, jeżdżące napisy i inne temu podobne, powstały w tym okresie. Przykładem jednej z niewielu stron z tamtego okresu, które wciąż działają, jest sławny w kręgach projektantów stron `https://www.arngren.net/` — niewielki norweski sklep internetowy. Zachęcam, by obejrzeć kilka stron i spróbować „się połapać".

Pokonana firma Netscape wykonała ruch, który początkowo został zignorowany przez konkurencję. Był to dość nowy pomysł jak na tamten czas — wiadomo było, że z Netscape Navigator nie będzie już zysków, więc firma udostępniła kody do swojej przeglądarki wszystkim na licencji open source. Firma założyła też fundację, której nazwa pochodzi od maskotki zespołu programistów pracujących nad Netscape Navigator. Był to czerwony pluszak przypominający dinozaura, którego nazywali Mozilla — z połączenia nazwy konkurenta (Mosaic) i sławnego japońskiego potwora (Godzilla). Na podstawie udostępnionego kodu opracowali przeglądarkę Phoenix, która zmieniła nazwę na Firebird, a potem ze względu na konflikt nazw z inną aplikacją, na Firefox.

W kwietniu 2004 roku Opera i Mozilla połączyły siły, aby opracować standardy sieci i posprzątać krajobraz po bitwie z pierwszej wojny przeglądarek. Okres ten uznaje się za początek drugiej wojny przeglądarek. Pół roku później Mozilla wydała Firefox 1.0. Od tego czasu Firefox systematycznie zyskiwał na popularności, zdobywając 32% rynku w 2009 roku. W 2008 roku do sceny dołączyła firma Google z przeglądarką Chrome. Do 2012 roku Chrome zrównał się popularnością z Firefoxem. Chrome ma obecnie 65.77% rynku, na drugim miejscu jest Edge z 12.71%, Safari z 8.63%, Firefox z 6.61% i Opera z 3.15%[3]. 25 maja 2017 roku prezes fundacji Mozilli ogłosił swoją porażkę w drugiej wojnie przeglądarek, co oznaczało jej koniec[4]. W drugiej wojnie przeglądarek nie chodziło już o jawne zyski, ponieważ przeglądarki są darmowe. Niemniej jednak okazuje się, że wciąż mogą generować duży przychód. Podczas gdy niektóre przeglądarki w standardzie dają możliwość blokowania reklam, inne oficjalnie deklarują, że nie będą wprowadzać blokad reklam, tylko planują

[3]Źródło: `https://gs.statcounter.com/browser-market-share/desktop/worldwide/#monthly-200901-202404`.

[4]Źródło: `https://andreasgal.com/2017/05/25/chrome-won/`.

pracować z dostawcami reklam, aby były lepiej nakierowane na odbiorcę i nie były uciążliwe[5]. Daje to wciąż dużą możliwość zysku, ponieważ nie wszyscy wiedzą, że mogą doinstalować blokadę reklam w formie wtyczki. Dodatkowo wiele przeglądarek zbiera informacje, które mogą być przydatne dla reklamodawców, aby lepiej ukierunkować reklamowane treści. Niezależnie od zwycięzcy, krajobraz po bitwie oczyścił się pod względem standardów i Internet jest o wiele bardziej estetycznym miejscem, choć prywatność jego użytkowników stoi pod znakiem zapytania.

Istnieje możliwość, że nadciąga trzecia wojna przeglądarek i fundacja Mozilli ponownie planuje wziąć w niej udział. Krótko po przegraniu drugiej wojny przeglądarek ogłoszono projekt Quantum. Projekt ten prowadzony był przez długi czas i polegał na przepisywaniu fragmentów przeglądarki na nowy język programowania — Rust. Rust powstał w fundacji Mozilli i jest jednym z pierwszych języków, które posiadają możliwość wyłapania większości błędów związanych z zarządzaniem pamięcią na etapie kompilacji, jednocześnie dając wydajność porównywalną z językami C lub C++. Język ten eliminuje praktycznie możliwość powstania błędów takich jak sławny Hearthbleed[6] — przy czym według danych 70% błędów, w tym te odpowiedzialne za utratę danych lub brak ich prywatności, jest spowodowane właśnie problemami w zarządzaniu pamięcią[7]. Między innymi z tego powodu Rust od 2017 roku (przez 6 lat z rzędu) jest najbardziej lubianym językiem programowania[8]. Jeśli nic się nie zmieni, ważnym aspektem Trzeciej Wojny Przeglądarek będzie prywatność[9], otwartość i równość[10]. Dla osób obserwujących dwie wcześniejsze wojny, obecny rozwój wydarzeń jest bardzo interesujący.

2 Standardy dziś

Na rynku obecnie znajduje się dużo przeglądarek. Z łatwością można znaleźć ponad 50 aktywnych przeglądarek. W rzeczywistości jednak są to różne interfejsy do tych samych silników renderujących. Silnik renderujący jest sercem przeglądarki, który zajmuje się wyświetlaniem kodu HTML oraz CSS. Obecnie aktywne i popularne silniki renderujące oraz silniki JavaScript, służące jako podstawa do różnych aplikacji internetowych, to:

1. **Blink/V8**, wykorzystywany w przeglądarkach Chrome (od wersji 28), Edge (od wersji 79), Opera (od wersji 15), Vivaldi, Brave, Amazon Silk, Avast Browser, Samsung Internet i wielu innych,

[5] Źródło: https://www.cnet.com/news/google-chrome-no-build-ad-blocker-advertising/?ftag=COS-05-10aaa0b&linkId=31033645.

[6] Jeden z niewielu błędów, który ma swoją nazwę i logo oraz stronę: https://heartbleed.com/.

[7] Źródło: https://www.zdnet.com/article/microsoft-70-percent-of-all-security-bugs-are-memory-safety-issues/.

[8] Źródło: https://survey.stackoverflow.co/2023/#section-admired-and-desired-programming-scripting-and-markup-languages.

[9] O to można zadbać już teraz, wybierając alternatywne narzędzia dbające o prywatność użytkowników, na przykład katalogowane na liście https://www.privacytools.io/.

[10] Źródło: https://www.mozilla.org/en-US/about/manifesto/.

2. **Webkit/JavaScriptCore**, wykorzystywany w przeglądarce Safari, mobilnych przeglądarkach na systemie iOS, przeglądarce LG Smart TV i kilku innych,

3. **Gecko/SpiderMonkey**, wykorzystywany w przeglądarce Firefox i kilku innych.

Aby przetestować jakąś stronę internetową, warto wybrać po jednej przeglądarce z każdego silnika i sprawdzić, jak na niej działa. Dzięki temu uzyskamy bardzo duże pokrycie możliwości.

Wszystkie wymienione tu silniki dobrze obsługują współczesne standardy. Na zajęciach będziemy wykorzystywać niewielki podzbiór standardów:

1. **HTML – żywy standard**, dawniej HTML5,

2. **CSS, wersja 2023**, dawniej CSS3,

3. **ECMAScript 2023**, dawniej JavaScript.

Obecnie odchodzi się od numerów standardów. Standardy tu wypisane po drugiej wojnie przeglądarek na tyle się przyjęły, że numerowanie ich wersjami straciło sens i w zamian, prowadzi się ciągły ich rozwój i co rok publikuje nową wersję z dotychczasowymi zmianami i udoskonaleniami. Nie są to jednak zmiany rewolucyjne, tylko ewolucyjne, stąd decyzja o zaprzestaniu numerowania standardu HTML po wersji 5.2 oraz standardu CSS. Wszystkie te języki są używane w tak zwanej fasadzie (ang. frontend). Inaczej mówiąc, po stronie klienta, czyli są uruchamiane przez przeglądarkę.

Aby strona internetowa działała, oprócz fasady potrzeba też oprogramowania działającego po stronie serwera. Podczas zajęć użyjemy kilku technologii, do najważniejszych zaliczać będziemy:

1. **micro-framework Flask**, do zaprogramowania zachowania strony internetowej po stronie serwera w języku Python,

2. **biblioteka Jinja**, do przetwarzania szablonów stron.

Wszystkie pięć opisanych tu technologii w sumie pozwoli nam stworzyć prostą stronę internetową. Osoby, które wybiorą kurs z baz danych w kolejnych semestrach, nauczą się o brakującej nam technologii, aby tworzyć bardziej zaawansowane systemy.

3 Podstawy języka HTML

HTML to język, który służy do opisu treści dokumentu. W HTML nie opisujemy jeszcze stylu, zatem strona wykonana tylko w HTML będzie wyglądać jak „napisane czarno na białym". Omówimy tu tylko najważniejsze elementy jego składni. W skład języka wchodzą

- znaczniki (inaczej tagi), stanowiące elementy lub grupujące elementy dokumentu,

- atrybuty, które pozwalają ustalić parametry przypisane znacznikom,

- encje, które pozwalają zapisać znaki, których nie ma na klawiaturze lub mają specjalne znaczenie,
- deklaracje, które nie stanowią treści dokumentu, tylko są informacją pomocniczą dla czytelnika kodu lub przeglądarki.

Pomiędzy LaTeXem a HTMLem jest wiele podobieństw, dlatego przy różnych okazjach będą one podkreślane.

Znaczniki dzielimy na puste i zwykłe. Znaczniki puste nie otaczają treści. Zaczynamy je od znaku <, kończymy znakiem >. W środku umieszczamy słowo będące nazwą tagu. Przykładowym znacznikiem pustym jest `<hr>` wstawiające linię poziomą. Zwykłe znaczniki otaczają część dokumentu i mogą mieć w sobie inne znaczniki. Znacznik zwykły zaczynamy tak samo jak znacznik pusty, natomiast kończymy znacznikiem zamykającym rozpoczynającym się od </ i kończącym na >. Przykładem znacznika zwykłego jest „p" oznaczający akapit.

```
<p>Treść akapitu.</p>
```

Tagi w tej formie są podstawowym elementem budulcowym języka HTML. O znacznikach pustych możemy myśleć jak o komendach LaTeXa, natomiast o znacznikach zwykłych jak o środowiskach.

Znaczniki możemy podzielić też na sposób wyświetlania. Niektóre znaczniki są znacznikami liniowymi. Znaczniki liniowe zachowują się jak litery. Przykładem znacznika liniowego jest `<em>...</em>`, oznaczający wyróżnienie. Innym typem znaczników są znaczniki blokowe, które powodują zakończenie obecnej linii. Przykładem znacznika blokowego jest akapit. Różnica pomiędzy znacznikami liniowymi a blokowymi jest podobna jak pomiędzy wzorami matematycznymi w linii w LaTeXu a wzorami wyróżnionymi złożonymi za pomocą środowiska `equation`.

Każdy znacznik może mieć ustalone atrybuty. Atrybuty dzielimy na dwa rodzaje — atrybuty z wartościami i atrybuty bez wartości. Atrybuty z wartościami wypisujemy jako pary klucz-wartość z symbolem równości pomiędzy, na przykład

```
<p lang=”en” class=”famous-quote”>To be or not to be!</p>
```

zawiera dwa atrybuty z wartościami. Pierwszy atrybut to `lang` ustawiony na wartość `”en”`, natomiast drugi to atrybut `class` ustawiony na wartość `”famous-quote”`. Jeśli jakiś atrybut ma wartość `””`, nazywamy go atrybutem bez wartości i zapisujemy podając jedynie jego nazwę. Oznacza to, że

```
<input required=””>
```

oraz

```
<input required>
```

są równoważne.

Do zapisu niektórych znaków musimy użyć encji. Encje zaczynamy znakiem `&` i kończymy znakiem `;`. Do najpopularniejszych encji należą:

- `<` oznaczający znak <, który jest częścią składni znaczników,
- `>` oznaczający znak >, który jest częścią składni znaczników,
- `&` oznaczający znak &, który jest częścią składni encji,
- ` ` oznaczający niełamliwą spację (jak ~ w LaTeXu),
- `&mdash;` oznaczający pauzę — (jak `\pauza` w LaTeXu),
- `&ndash;` oznaczający półpauzę – (jak `\ppauza` w LaTeXu),
- `&laquo;` oznaczający otwierający cudzysłów francuski « (jak << w LaTeXu),
- `&raquo;` oznaczający zamykający cudzysłów francuski » (jak >> w LaTeXu),
- `&bdquo;` oznaczający otwierający cudzysłów polski „ (jak `,,` w LaTeXu),
- `&rdquo;` oznaczający zamykający cudzysłów polski ” (jak `''` w LaTeXu),
- `&hellip;` oznaczający wielokropek …(jak `\ldots` w LaTeXu),
- `&euro;` oznaczający symbol waluty euro € (jak `\euro` w LaTeXu),
- `&copy;` oznaczający symbol praw autorskich innych niż muzyczne © (jak `\copyright` w LaTeXu).

Więcej encji możemy znaleźć na przykład na stronie `http://www.amp-what.com/`.

Ostatnim nieopisanym jeszcze elementem języka HTML są deklaracje. Deklaracje rozpoczynamy od symbolu `<!` i kończymy symbolem >. Wyróżniamy dwa popularne wykorzystania deklaracji:

1. deklaracja `<!DOCTYPE html>` na początku dokumentu, informująca o typie dokumentu (jak `\documentclass` w LaTeXu),
2. komentarze `<!-- komentarz -->`, które są ignorowane przez przeglądarkę (jak `%` w LaTeXu).

Każdy prawidłowy dokument powinien zaczynać się deklaracją typu i mieć jeden tag `html`. Wewnątrz powinny być dwa tagi — `head` i `body`. Pełnią one podobną funkcję jak preambuła i ciało dokumentu w LaTeXu. Minimalny dokument w standardzie HTML5 lub nowszym, powinien zawsze mieć

```
<!DOCTYPE html>
<html>
<head>

</head>
<body>

</body>
</html>
```

Powinniśmy w dokumencie takim ustawić również język za pomocą atrybutu `lang`. Aby ustawić język dla całego dokumentu, należy ustawić go w tagu `html`. Dodatkowo w tagu `head` powinniśmy ustawić kodowanie znaków oraz tytuł. W tagu `body` umieszczamy treść.

```
<!DOCTYPE html>
<html lang="pl">
<head>
<meta charset="utf-8">
<title>Moja pierwsza strona &mdash; hura!</title>
</head>
<body>
<h1>Nagłówek</h1>
<p>Witaj świecie!</p>
<!-- komentarz -->
</body>
</html>
```

Zawartość tego pliku tekstowego możemy zapisać w pliku o rozszerzeniu `.html` i otworzyć w przeglądarce, dwa razy w niego klikając.

4 Najważniejsze znaczniki HTML

Do najważniejszych narzędzi formatowania należą nagłówki, akapity, wyróżnienia i łącza:

- nagłówki to znaczniki od `<h1>...</h1>` do `<h6>...</h6>` oznaczające kolejne poziomy tytułów (jak `chapter`, `section`, `subsection`, …w LaTeXu);
- akapity to znaczniki `<p>...</p>` oznaczające akapity;
- wyróżnienia czyli znaczniki `em` oraz `strong` odpowiadające dwóm poziomom wyróżnienia;

- hiperłącza czyli linki `<a href=”adres”>tekst</a>`. Hiperłącza wyświetlają tekst, który po kliknięciu przekierowuje przeglądarkę na stronę wskazaną przez adres w atrybucie `href`.

Przyjrzyjmy się teraz hiperłączom. Załóżmy, że jesteśmy na stronie

```
https://moja.strona/user/profile/address.html
```

Przypomnijmy elementy budowy adresu w standardzie URL. W powyższym adresie:

- protokół to `https`,
- domena to `moja.strona`,
- ścieżka do zasobu to `/user/profile/address.html`.

W ścieżce do zasobu widać analogię do struktury katalogów: `/home/student/plik.txt` jest ścieżką do pliku `plik.txt` znajdującego się w katalogu `/home/student`. W przypadku naszego adresu, mamy zasób `address.html` znajdujący się w katalogu `/user/profile`.

Załóżmy teraz, że edytujemy plik `address.html`, który opublikujemy jako

```
https://moja.strona/user/profile/address.html
```

Chcemy w nim odwołać się do banera pod adresem `https://moja.strona/img/baner.png`, używając atrybutu `href`. Możemy to zrobić na cztery sposoby:

- bezwzględnie, zaczynając od protokołu, czyli `href=”https://moja.strona/img/baner.png”`;
- względnie do protokołu, zaczynając od dwóch znaków `//`, czyli `href=”//moja.strona/img/baner.png”`;
- względnie do domeny, zaczynając od jednego znaku `/`, czyli `href=”/img/baner.png”`;
- względnie do zasobu, zaczynając od katalogu, w którym znajduje się zasób, czyli `href=”../../img/baner.png”`.

Jeśli uruchamiamy stronę lokalnie, warto wykorzystać względne odwoływanie się do innych zasobów.

Ilustracje na stronie załączamy za pomocą tagu `<img src=”plik” alt=”tekst”>`. Jest to pusty znacznik, czyli nie musimy go zamykać. Musimy natomiast podać przynajmniej dwa atrybuty — `src` wskazujący na załączany plik oraz `alt` opisujący tekst dla osób, które nie mogą zobaczyć obrazka, czyli na przykład niewidomych użytkowników naszej strony, korzystających z czytników ekranowych. Ścieżki podajemy tak samo jak w atrybucie `href` w hiperłączach, czyli bezwzględnie lub względnie do domeny albo dokumentu.

Podobnie jak w LaTeXu mamy trzy sposoby na tworzenie list:

- wypunktowania tworzymy, umieszczając jedną lub więcej pozycji `<li>...</li>` wewnątrz `<ul>...</ul>` (`li` pochodzi od „list item" czyli „element listy", `ul` od „unordered list" czyli „lista nieuporządkowana");
- wyliczenia tworzymy, umieszczając jedną lub więcej pozycji `<li>...</li>` wewnątrz `<ol>...</ol>` (`ol` pochodzi od „ordered list" czyli „lista uporządkowana");
- definicje tworzymy, umieszczając jedną lub więcej pozycji `<dt>...</dt>` i `<dd>...</dd>` wewnątrz `<dl>...</dl>` (`dt` pochodzi od „definition term", `dd` od „definition description", `dl` od „definition list" czyli kolejno wyrażenie, opis i lista definicji).

Poniżej przykłady wypunktowań w HTML.

```
<ul>
<li>podpunkt,</li>
<li>podpunkt.</li>
</ul>
<ol>
<li>podpunkt 1,</li>
<li>podpunkt 2.</li>
</ol>
<dl>
<dt>red</dt><dd>czerwony,</dd>
<dt>blue</dt><dd>niebieski.</dd>
</dl>
```

Wypunktowania nie powinny znajdować się wewnątrz akapitów, ponieważ same w sobie są znacznikami blokowymi.

Tabele definiujemy za pomocą znacznika `<table>...</table>`. Wewnątrz tego znacznika musimy użyć przynajmniej jednego rzędu `<tr>...</tr>` a w nim przynajmniej jednej komórki zwykłej `<td>...</td>` lub komórki nagłówkowej `<th>...</th>`. Komórki nagłówkowe domyślnie są wyśrodkowane i złożone pogrubionym krojem pisma. Spróbujmy złożyć poniższą tabelę.

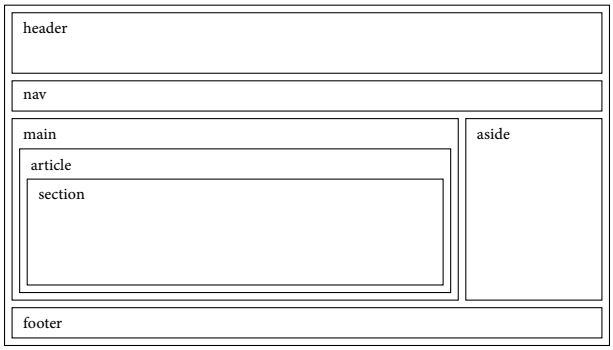
Numer indeksu	**Ocena**
123456	5.0
654321	3.5

Tabelę taką uzyskamy kodem

```
<table>
<tr>
<th>Numer indeksu</th>
<th>Ocena</th>
</tr>
<tr>
```

```
<td>123456</td>
<td>5.0</td>
</tr>
<tr>
<td>654321</td>
<td>3.5</td>
</tr>
</table>
```

W HTML 5 wprowadzono znaczniki strukturalne, które pozwalają wskazać znaczenie różnych fragmentów strony. Na ich podstawie między innymi czytniki ekranowe wiedzą, jak nawigować po treści osoby niewidome. Poniżej przykład ułożenia tych elementów.



Najważniejsze tagi strukturalne to:

- `<header>...</header>` — najczęściej nagłówek strony z jej nazwą,
- `<nav>...</nav>` — nawigacja, czyli menu do poruszania się po stronie,
- `<main>...</main>` — główna część strony,
- `<article>...</article>` — najczęściej treść dzieli się na artykuły (jak w blogu lub forum),
- `<section>...</section>` — artykuł lub cała strona może być podzielona na sekcje,
- `<aside>...</aside>` — często jest też pasek boczny na przykład z archiwum i wyszukiwarką,
- `<footer>....</footer>` — w stopce zwykle umieszczane są informacje o autorach i prawach autorskich, dane kontaktowe i temu podobne.

Aby na kolejnych etapach mieć możliwość korzystania z wielu dobrodziejstw języków CSS oraz ECMAScript, musimy móc wskazać fragmenty dokumentów HTML wewnątrz pliku CSS z opisem stylu oraz wewnątrz skryptu w ECMAScript.

Do tego celu stosuje się atrybuty `class` oraz `id`. Atrybut `id` powinien być unikatowy wewnątrz całego pliku HTML. Mamy również tagi `<span>...</span>` oraz `<div>...</div>`, które są znacznikami „nic-nie-robiącymi". Ich jedynym zastosowaniem jest pogrupowanie innych znaczników lub fragmentów tekstu w kontener, do którego możemy dodać atrybuty `id` lub `class`. `span` jest atrybutem liniowym a `div` jego odpowiednikiem blokowym.

5 Walidacja HTML

Aby kod wyświetlał się poprawnie na wszystkich głównych przeglądarkach, powinien być zgodny ze standardami. Istnieje narzędzie, które pomaga automatycznie sprawdzić, czy strona jest zgodna ze standardami. Narzędzia sprawdzające poprawność kodu w świetle standardów nazywamy walidatorami. Walidator HTML jest dostępny online na stronie `https://validator.w3.org/nu/`. Aby z niego skorzystać, możemy wkleić treść strony w odpowiednie okno, przesłać plik lub podać adres strony w Internecie.

Jeśli umieścimy już stronę w Internecie, możemy też sprawdzić plik HTML przy pomocy walidatora dostępności, sprawdzającego jak dostępna dla osób z ograniczeniami jest dana strona. Można to zrobić pod adresem `https://wave.webaim.org/`.

W razie wątpliwości zawsze można sprawdzić informacje w standardzie, dostępnym za darmo na stronie `https://html.spec.whatwg.org/multipage/`. Przydatnym linkiem jest również dokumentacja na stronie fundacji Mozilli `https://developer.mozilla.org/en-US/docs/Learn/HTML/Introduction_to_HTML` oraz dokumentacja na stronie W3Schools `https://www.w3schools.com/tags/`.