Web-технології та web-дизайн. CSS

2024/2025

Самостійна робота
до лекції №3

Лекції: ст. викладач каф. Штучного інтелекту
Гриньова Олена Євгенівна
olena.hrynova@nure.ua

CSS Комбінатори

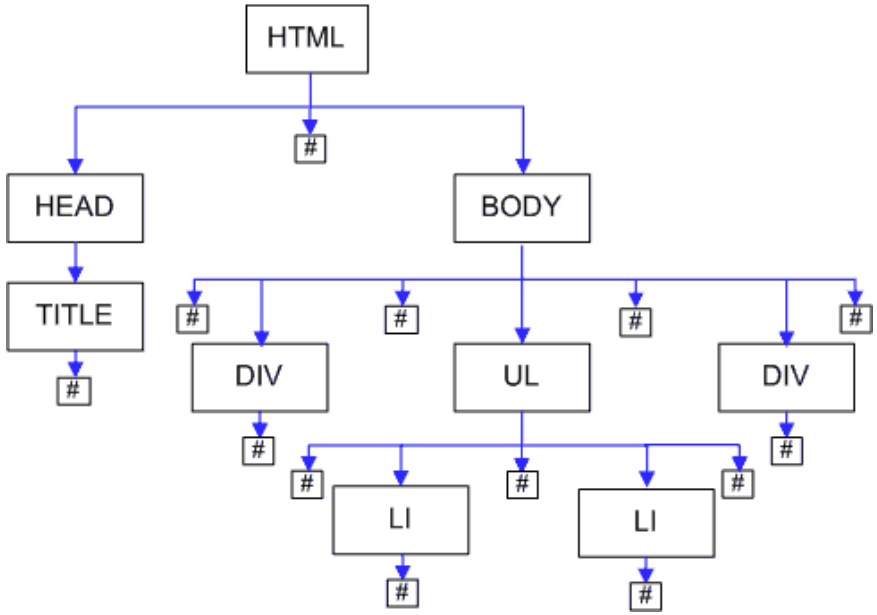
DOM

CSS комбінатори використовуються для пошуку та вибору елементів по об'єктному дереву HTML-документа (DOM) з метою їхньої стилізації.
HTML-документ структурований як дерево, де кожен елемент (наприклад, , <p>, <h1>, і т.д.) є вузлом у цьому дереві. Відносини між цими вузлами, такі як батьківські і дочірні елементи, брати і нащадки, визначають, як елементи пов'язані між собою.

DOM

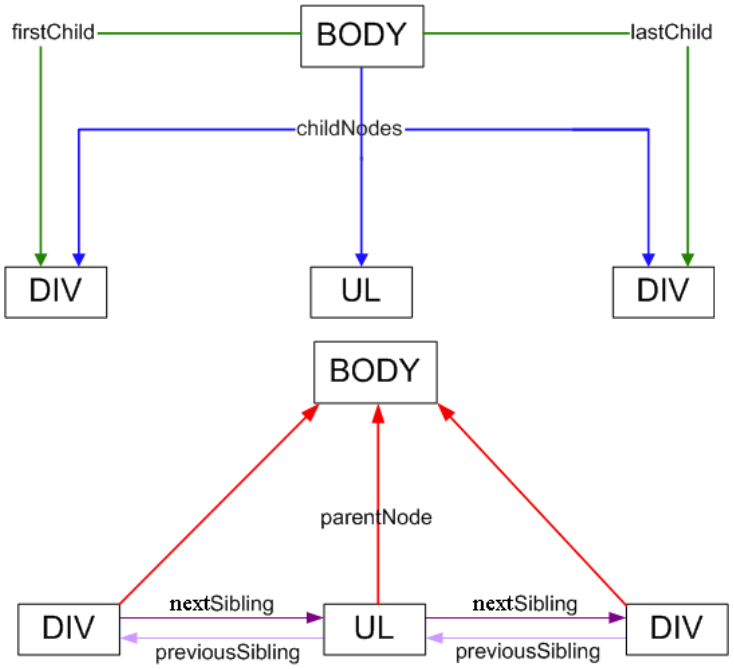
Об'єктна модель документа (**DOM**) — програмний інтерфейс (API) для документів HTML і XML.
DOM забезпечує структуроване представлення документа і визначає, як до цієї структури можна отримати доступ із програм, які можуть змінювати вміст, стиль і структуру документ.

```
<html>
  <head><title>...</title></head>
  <body>
    <div id="dataKeeper">Data</div>
    <ul>
      <li style="color:red">Обережно,</li>
      <li class="info">Інформація</li>
    </ul>
    <div id="footer">Made in Ukraine;</div>
  </body>
</html>
```



DOM

Об'єктна модель документа (**DOM**) — програмний інтерфейс (API) для документів HTML і XML.
DOM забезпечує структуроване представлення документа і визначає, як до цієї структури можна отримати доступ із програм, які можуть змінювати вміст, стиль і структуру документ.




```
<html>
 <head><title>...</title></head>
 <body>
   <div id="dataKeeper">Data</div>
   <ul>
     <li style="color:red">Обережно,</li>
     <li class="info">Інформація</li>
   </ul>
   <div id="footer">Made in Ukraine &copy;</div>
 </body>
</html>
```

CSS Комбінатори

Розглянемо селектори, які називаються **комбінаторами**, оскільки вони з'єднують інші селектори, створюючи корисний зв'язок селекторів один з одним та розташуванням вмісту в документі.

CSS селектор може містити більш ніж однин простий селектор. Між простими селекторами можна вставити **комбінатор**.

В CSS є чотири різні **комбінатора** :

- **селектор нащадків** (пробіл)
- **селектор дочірній** (>)
- **селектор сусідній споріднений** (+)
- **селектор загальний споріднений** (~)

CSS комбінатори. Комбінатор нащадка

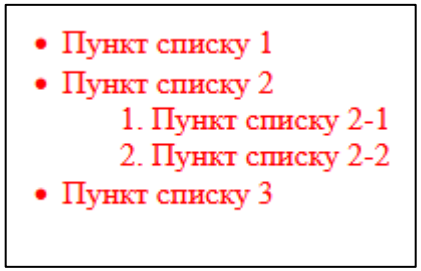
Комбінатор нащадка / селектор нащадка / контекстні селектори / вкладені селектори — зазвичай являє собою символом пробілу ( ) — з'єднує два селектори так, що елементи, що відповідають другому селектору, вибираються, **якщо вони мають предка** (батька, батька батька, батька батька батька і т.д.), що відповідає першому селектору.

Селектори, що використовують комбінатор нащадка, називаються селекторами нащадка.

Комбінатор нащадку

HTML

```
<ul>
  <li>Пункт списку 1</li>
  <li>Пункт списку 2
    <ol>
      <li>Пункт списку 2-1</li>
      <li>Пункт списку 2-2</li>
    </ol>
  </li>
  <li>Пункт списку 3</li>
</ul>
```



CSS

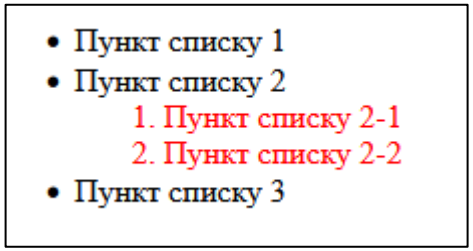
```
ul li {color: red;}
```

Функціонування **селектора нащадків**, ґрунтується на його родинних відносинах.

Комбінатор нащадку

HTML

```
<ul>
  <li>Пункт списку 1</li>
  <li>Пункт списку 2
    <ol>
      <li>Пункт списку 2-1</li>
      <li>Пункт списку 2-2</li>
    </ol>
  </li>
  <li>Пункт списку 3</li>
</ul>
```



CSS

```
ol li {color: red;}
```

Селектори нащадків

Селектори нащадків - поєднання простоти використання селекторів тегів з точністю селекторів класів та ідентифікаторів.

Селектори нащадків застосовують для того, щоб однаково відформатувати цілий набір тегів, але **тільки коли вони розташовані у певному контексті – у фрагменті вебсторінки (розміщені всередині інших елементів).**

Селектори нащадків

```
<!DOCTYPE html>
<html>
 <head>
   <meta charset = "UTF-8">
   <title>Селектори нащадків </title>
   <style>
     h2 i {color:red; /* задаємо колір шрифту */}
     p i {color:green; /* задаємо колір шрифту*/}
   </style>
 </head>
 <body>
    <h2>Заголовок <i>другого</i> рівня</h2>
    <h2>Заголовок другого рівня</h2>
    <p>text <i>text1</i> <b>text2</b> text</p>
    <p>text text1 <i>text1</i> text2<i> text3</i> </p>
 </body>
</html>
```

Заголовок *другого* рівня

Заголовок другого рівня

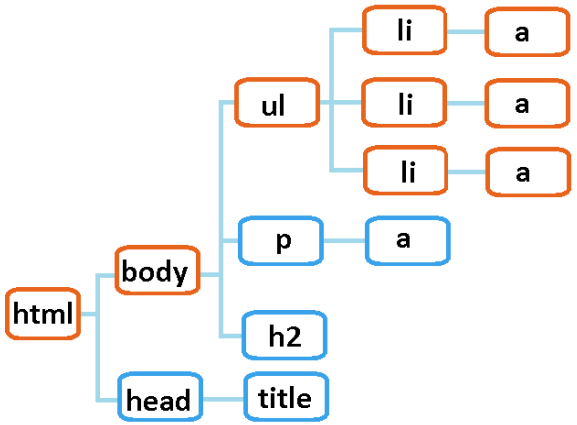
text *text1* **text2** text

text text1 *text1* text2 *text3*

Тобто зелений колір шрифту буде застосовуватися тільки до тих елементів **i**, які знаходяться всередині елемента **p**, а червоний колір шрифту буде застосовуватися тільки до тих елементам **i**, які знаходяться всередині елемента **h2**

Селектори нащадків

Селектори нащадків дозволяють використовувати **дерево HTML**, форматуючи теги по-різному, залежно від того, де вони розташовані.
Селектори нащадків стилізують вкладені елементи вебсторінки, дотримуючись тих самих правил, яким підкоряються **теги-предки** і **теги-нащадки** в **дереві HTML**.
Ви створюєте **селектор нащадків**, об'єднуючи селектори разом (відповідно до гілки дерева, яку потрібно відформатувати), поміщаючи **найстаршого предка зліва,** а форматований тег - **справа**. Обов'язково розділяйте пробілами предків та нащадків.



html body ul li a {блок оголошень}
html ul li a {блок оголошень}
html li a {блок оголошень}
body ul li a {блок оголошень}
body li a {блок оголошень}
ul li a {блок оголошень}
li a {блок оголошень}

Селектори дочірніх елементів

Селектор дочірніх елементів використовує додатковий символ — кутову дужку (>) — для зазначення відношення між двома елементами.

Селектор 1 > Селектор 2 {Опис правил стилю }

Наприклад,

body > h1 - вибирає будь-який тег <**h1**>, дочірній по відношенню до <**body**>.

На відміну від селектора нащадків, який застосовується до всіх нащадків (тобто вкладених елементів), **селектор дочірніх елементів дозволяє визначити конкретні дочірній та батьківській елементи.**

Сусідні селектори

Сусідніми називаються елементи вебсторінки, коли вони йдуть безпосередньо один за одним у коді документа.

Синтаксис

E+F {Опис правил стилю }

Для керування стилем сусідніх елементів використовується символ плюса (+), який встановлюється між двома селекторами E та F. Пропуски навколо плюса не обов'язкові. **Стиль при такому записі застосовується до елемента F, але тільки в тому випадку, якщо він є сусіднім елементу E і розташовується відразу після нього.**

Сусідні селектори

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="utf-8">
  <title> Сусідні селектори </title>
  <style>
   b + i {
    color: red; /* Червоний колір тексту */
   }
  </style>
 </head>
 <body>
   <p>Lorem <b>ipsum </b> dolor sit amet, <i>consectetuer</i> adipiscing elit.</p>
   <p>Lorem ipsum dolor sit amet, <i>consectetuer</i> adipiscing elit.</p>
 </body>
</html>
```

Lorem **ipsum** dolor sit amet, *consectetuer* adipiscing elit.

Lorem ipsum dolor sit amet, *consectetuer* adipiscing elit.

Споріднені селектори

Споріднені селектори за своєю поведінкою схожі на сусідні селектори (запис виду E + F), але на відміну від них стильові правила застосовуються до всіх прилеглих елементів.

E~F {Опис правил стилю }

Для керування стилем споріднених елементів використовується символ тильди (~), він додається між двома селекторами E та F. Пропуски навколо тильди не обов'язкові. Стиль при такому записі застосовується до елемента F у тому випадку, якщо він має того ж батька, що і елемент E і розташовуються відразу після нього.

Як стилізувати всіх предків елемента?

У CSS немає прямого способу стилізувати всіх предків (ancestor elements) певного елемента. CSS селектори дозволяють вибирати елементи на основі їхнього положення серед нащадків, сусідів і загальних братів, але не предків. Однак є кілька обходних шляхів:

1. За допомогою JavaScript можна додати клас або стилі до предків елемента.
2. Використання CSS-in-JS або інших CSS-препроцесорів, можна динамічно генерувати класи або стилі для предків залежно від структури DOM.

3. Вплив на предків через стилізацію нащадків

Хоча це не стилізація предків напряму, ви можете опосередковано вплинути на зовнішній вигляд предків, змінивши стилі

Обмеження CSS

У CSS є кілька речей, які неможливо стилізувати напряму або мають обмежену підтримку через специфіку роботи з DOM або відсутність відповідних селекторів чи властивостей. Ось кілька прикладів:

1. Батьківський елемент (предок)Як згадувалося раніше, стилізувати батьківський елемент напряму на основі нащадків (за виключенням випадків із :has() або :focus-within) не можна. Це обмеження стосується самої структури CSS, який працює зверху вниз — тобто стилі застосовуються до елементів на основі їхньої структури або відносин із сусідами, але не з їхніми батьками.

Обмеження CSS

2. Попередні елементи (сиблінги)CSS-селектори дозволяють стилізувати сусідні елементи, що йдуть після певного елемента (через селектори + та ~), але немає способу стилізувати попередні елементи, які знаходяться перед певним елементом. Тобто неможливо вибрати і стилізувати елемент, який знаходиться перед іншим елементом.
3. Стихійні елементи за станом кількох іншихCSS не дозволяє напряму стилізувати елемент на основі стану кількох інших елементів. Наприклад, неможливо створити правило, за якого елемент змінював би свої стилі лише тоді, коли кілька інших елементів одночасно мають певний клас або властивість.

Обмеження CSS

4. Висоту або ширину, засновану на вмісті нащадкаCSS не дозволяє напряму зробити батьківський елемент змінювати свої розміри залежно від вмісту дочірнього елемента, крім випадків, коли використовується властивість display: flex; або display: grid;. Але для звичайних блокових елементів динамічне підстроювання батьків за контентом нащадків не працює без спеціальних трюків.
5. Складні математичні обчисленняCSS має обмежену підтримку математичних операцій, таких як додавання або віднімання значень у властивості calc(), але для більш складних обчислень (наприклад, процентних співвідношень між елементами або умовних обчислень) потрібно використовувати JavaScript.

Обмеження CSS

6. Перемикання стилів за станом елемента (за допомогою умов)CSS не підтримує умовні конструкції, такі як if-else, що є у програмуванні. Це означає, що неможливо змінювати стилі напряму залежно від певної умови (окрім базових станів, таких як :hover, :active, :focus). Для умовних стилів потрібно використовувати JavaScript.

7. Стилізація елементів за контентом (динамічні елементи)Хоча псевдоклас :empty дозволяє стилізувати порожні елементи, CSS не може перевіряти конкретний текстовий вміст елемента для його стилізації. Наприклад, не можна змінити колір тексту залежно від того, чи містить елемент певне слово або число. Для цього потрібні JavaScript або інші методи.

Обмеження CSS

8. Стилізація текстових вузлів напряму. Текстові вузли (окремі частини тексту без тегів) не можна стилізувати без обгортання їх в HTML-теги. CSS дозволяє стилізувати тільки елементи HTML, а для текстових фрагментів потрібно створювати додаткові елементи або використовувати псевдоелементи.
9. Елементи за межами DOM (наприклад, тіні iframe)Елементи, які знаходяться поза межами основного DOM-дерева (наприклад, елементи всередині iframe або інших вбудованих документів), не можна стилізувати з основного документа напряму за допомогою CSS.

Обмеження CSS

10. Псевдоелементи за станом їхнього батькаПсевдоелементи, такі як ::before і ::after, створюють додатковий контент перед або після елементів, але не можна стилізувати ці псевдоелементи на основі стану їхнього батька або інших елементів.

Висновок:CSS має свої обмеження через те, що він здебільшого описує **декларативну** систему стилів, яка працює з елементами **статично**, не реагуючи на змінні умови або логіку без допомоги JavaScript. Однак нові специфікації та прийоми (такі як селектор :has(), використання CSS Grid та Flexbox) значно розширюють можливості.

Споріднені селектори

В HTML-документі елементи можуть знаходитися в різних відносинах один з одним, що впливає на те, як вони взаємодіють у структурі сторінки. У CSS комбінатори допомагають визначати ці відносини, щоб застосувати стилі до елементів залежно від їхнього розташування відносно інших елементів.

Комбінатори допомагають створювати більш складні та точні селектори, що дозволяє гнучко керувати стилями залежно від структури HTML-документа.

Комбінатори використовуються для визначення способів комбінування або впорядкування елементів на веб-сторінці, щоб вибирати і стилізувати їх відповідно до їхньої позиції в DOM-дереві.

Література

On-line підручники та довідники:

- https://www.w3.org/TR/?title=css

Дякую за увагу!