Лекція 5
«Технологія CSS (частина 2)»

1. Блочна модель форматування CSS

From the People of Japan
UNDP Ukraine
«Frontend-developer: обучающая программа для переселенцев» реализуется в рамках Проекта Програмы развития ООН «Быстрое реагирование на социальные и экономические проблемы внутренне перемещённых лиц в Украине» при финансовой поддержке Правительства Японии.

Типи елементів

- ✔ Блочні
- ✔ Рядкові (строчні)
- ✔ Рядково-блочні

Виділяють окрему групу елементів, які браузер обробляє як рядково-блочні. Такі елементи є вбудованим, але для них можна задавати поля, відступи, ширину і висоту.

Блочные

- <address>...</address>,
- <article>...</article>,
- <aside>...</aside>,
- <body>...</body>,
- <blockquote>...</blockquote>,
- <dd>...</dd>,
- ... ,
- <dl>...</dl>,
- <dt>...</dt>,
- <fieldset>...</fieldset>,
- <figcaption>...</figcaption>,
- <figure>...</figure>,
- <footer>...</footer>,
- <form>...</form>,
- <h1>-<h6>...</h1>-</h6>,
- <header>...</header>,
- <hgroup>...</hgroup>,
- <hr>,
- ...,
- <legend>...</legend>,
- <nav>...</nav>,
- ...,
- <output>...</output>,
- ... ,
- <pre>...</pre>,
- <ruby>...</ruby>,
- <section>...</section>,
- ...

Строчные

- <a>...,
- <abbr>...</abbr>,
- <acronym>...</acronym>,
- <area>,
- ...,
- <bdo>...</bdo>,
-
,
- <cite>...</cite>,
- <code>...</code>,
- <dfn>...</dfn>,
- ...,
- ...,
- <img>,
- <kbd>...</kbd>,
- <label>...</label>,
- <map>...</map>,
- <noscript>...</noscript>,
- <object>...</object>,
- <samp>...</samp>,
- <script>...</script>,
- <small>...</small>,
- <source>,
- ...,
- ...,
- _{...},
- ^{...},
- <time>...</time>,
- <track>
- <q>...</q>,
- <u>...</u>,
- <var>...</var>

Строчно-блочные

- <button>...</button>
- <input>
- <keygen>...</keygen>
- <meter>...</meter>
- <progress>...</progress>
- <select>...</select>
- <textarea>...</textarea>

Блочеа модель елемента

- У блочній моделі елемент розглядається як прямокутний контейнер, що складається з:

✔ ***вмісту (content)***,

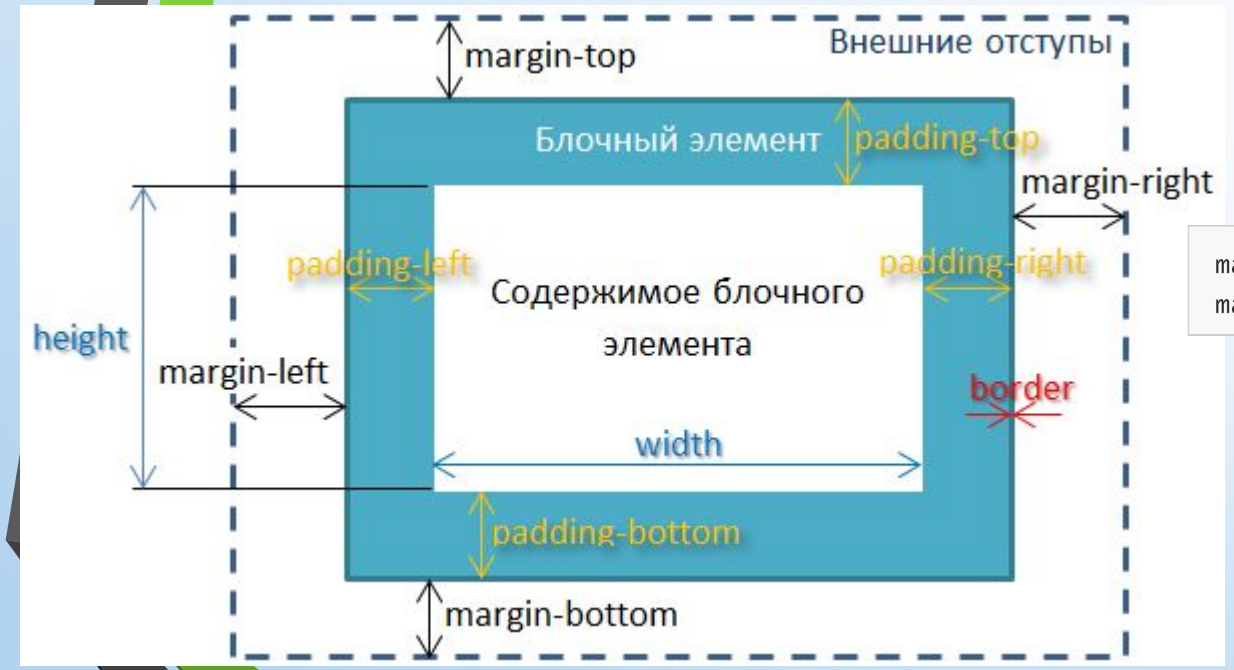
✔ ***внутрішнього відступу(padding)***,

✔ ***рамки (border)***,

✔ ***зовнішнього відступу (margin)***.

- **Область вмісту** - це вміст елемента, наприклад, текст або зображення.
- **Внутрішній відступ**, або поле елемента додає відступи всередині елемента, між його основним вмістом і його рамкою. Якщо для елемента задати фон, то він поширюється також і на поля елемента. Внутрішній відступ не може приймати від'ємних значень, на відміну від зовнішнього відступу.
- **Зовнішній відступ** додає відступи за межами елемента, створюючи тим самим проміжки між елементами. Вони завжди залишаються прозорими і через них видно фон батьківського елемента.
- **Рамка елемента** задається за допомогою властивості рамки. Якщо колір рамки не заданий, вона приймає колір основного вмісту елемента, наприклад, тексту. Якщо рамка має розриви, то крізь них буде проступати фон елемента.

Розміри блочних елементів



- Відповідно до блокової моделі загальна ширина елемента може бути розрахована за такою формулою:

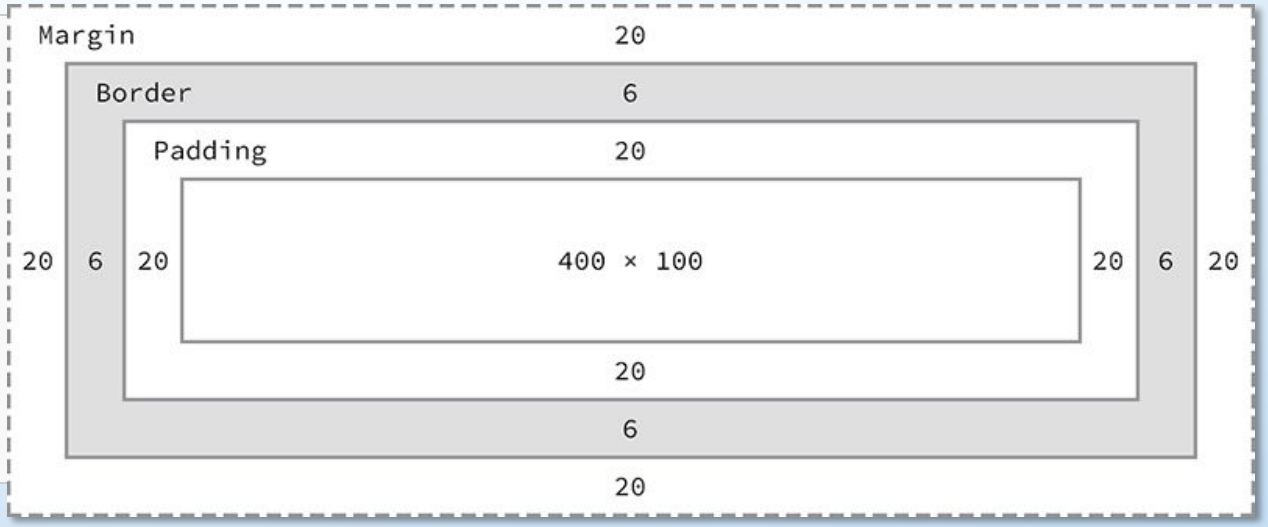
```
margin-right + border-right + padding-right + width + padding-left + border-left + margin-left
```

- Для порівняння, відповідно до блокової моделлі загальна висота елемента може бути розрахована за наступною формулою:

```
margin-top + border-top + padding-top + height + padding-bottom + border-bottom + margin-bottom
```

Приклад визначення розмірів блочних елементів

```
div {
  border: 6px solid #949599;
  height: 100px;
  margin: 20px;
  padding: 20px;
  width: 400px;
}
```




```
Ширина: 492px = 20px + 6px + 20px + 400px + 20px + 6px + 20px
Высота: 192px = 20px + 6px + 20px + 100px + 20px + 6px + 20px
```

Властивість **width** (ширина)

- За замовчуванням ширина елемента заснована на значенні display.
- У блочних елементів ширина за замовчуванням 100% і займає весь доступний горизонтальний простір.
- Рядкові та рядково-блочні елементи розширюються і стискаються горизонтально для розміщення їх вмісту.
- Рядкові елементи не можуть мати фіксований розмір, таким чином, ширина і висота відносяться тільки до НЕ рядкових елементів.
- Щоб задати певну ширину для НЕ рядкових елементів, використовуйте

```
div {
  width: 400px;
}
```

Властивість **height** (висота)

- Висота елемента за замовчуванням визначається його вмістом.
- Елемент буде розширюватися і стискатися по вертикалі при необхідності, щоб вмістити його вміст.
- Встановити певну висоту для НЕ рядкових елементів можна через властивість **height**:

```
div {
  height: 100px;
}
```

Властивість **margin** (зовнішній відступ)

- Властивість **margin** дозволяє нам встановити простір, яке оточує елемент.
- **margin** знаходяться за межами будь-яких кордонів і повністю прозорий в кольорі.
- Зовнішні відступи можуть використовуватися для позиціонування елементів в конкретному місці на сторінці або для додавання порожнього простору, зберігаючи всі інші елементи на певній відстані. Ось властивість **margin** в дії:

```
div {
  margin: 20px;
}
```

Властивість **padding** (внутрішній відступ)

- Властивість **padding** дуже схоже на властивість **margin**, проте розташовується всередині кордонів елемента.
- Властивість **padding** використовується, щоб задати простір безпосередньо всередині елемента.

```
div {
  padding: 20px;
}
```

Властивості **margin** і **padding** для рядкових елементів

- Рядкові елементи поводяться трохи по-іншому, ніж блокові і рядково-блочні елементи, коли справа доходить до відступів і полів.
- Для рядкових елементів **margin** працює тільки горизонтально - зліва і праворуч від елементів.
- **padding** працює на всіх чотирьох сторонах рядкових елементів.
- Відступи і поля працюють однаково для блочних і рядково-блочних елементів.

Форми позначення властивостей **margin** і **padding**

- При використанні скороченої властивості margin, щоб встановити одне значення для всіх чотирьох сторін елемента ми задаємо одне значення:

```
div {
  margin: 20px;
}
```

- Щоб встановити одне значення для верхньої і нижньої сторони, а інше значення для лівого і правого боку елемента, вкажіть два значення: спочатку зверху і знизу, потім ліворуч і праворуч.

```
div {
  margin: 10px 20px;
}
```

Тут для ми задаємо відступи 10 пікселів зверху і знизу і 20 пікселів зліва і справа.

Форми позначення властивостей **margin** і **padding**

- Щоб встановити унікальні значення для всіх чотирьох сторін елемента, вкажіть ці значення в наступному порядку: зверху, праворуч, знизу і зліва, рухаючись за годинниковою стрілкою.

```
div {
  margin: 10px 20px 0 15px;
}
```

Тут ми задаємо для відступи 10 пікселів зверху, 20 пікселів справа, 0 пікселів знизу і 15 пікселів зліва.

Форми позначення властивостей **margin** і **padding**

- У записі ми можемо встановити значення тільки для однієї сторони використовуючи унікальні властивості.
- Після імені кожної з властивостей (margin або padding в даному випадку) йде дефіс і сторона блоку, до якої має застосовуватися значення: top, right, bottom або left.
- Наприклад, властивість **padding-left** приймає тільки одне значення і встановить ліве поле для цього елемента, властивість **margin-top** приймає тільки одне значення і встановить верхній відступ для цього елемента.

```
div {
  margin-top: 10px;
  padding-left: 6px;
}
```

Форми позначення властивостей **margin** і **padding**

- Значення **padding** і **margin** задаються в наступному порядку: верхнє, праве, нижнє і ліве.

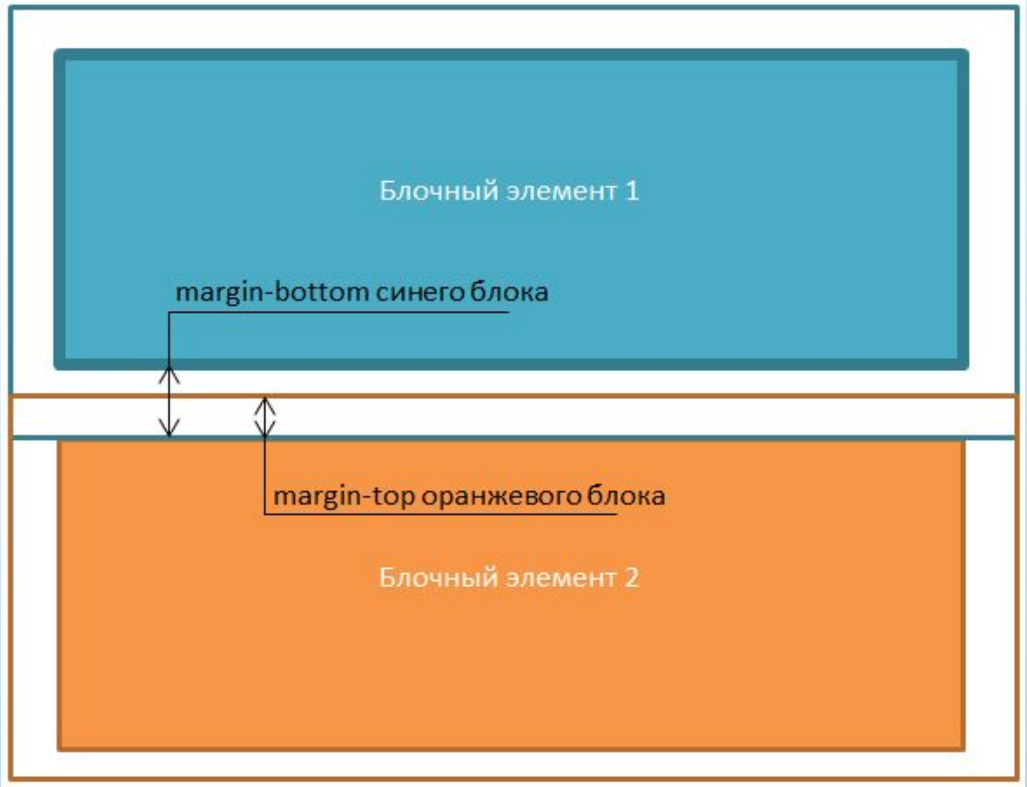
«Зхлопування» вертикальних відступів блокових елементів

- Коли два або більше вертикальних **margin** (**margin-top** і **margin-bottom**) стикаються, вони зливаються, при цьому ширина загального відступу дорівнює ширині більшого з вихідних відступів.
- Злиття виконується тільки для блочних елементів в нормальному потоці документа.
- Зовнішні вертикальні відступи рядкових, плаваючих і абсолютно позиціонованих елементів не зливаються.

Щоб отримати бажаний проміжок, можна задати, наприклад, для верхнього елементу **padding-bottom**, а для нижнього елемента -

«Зхлопування» вертикальних відступів блокових елементів

- Якщо серед відступів є негативні значення, то браузер додасть від'ємне значення до позитивного, а отриманий результат і буде відстанню між елементами.



Центрування блокових елементів:

✔ {margin-left: auto; margin-right: auto;}

✔ {margin: auto;}

2. CSS рамка

Рамки (border)

- Межі розташовуються між відступами і полями, створюючи рамку навколо елемента.
- Для властивості **border** потрібно три значення: ширина, стиль і колір.
- Скорочений запис для **border** задається цьому ж порядку - ширина, стиль, колір.
- У звичайному записі ці три значення можуть бути розбиті за властивостями **border-width**, **border-style** і **border-color**.
- Звичайний запис корисний для зміни або переписування окремого значення кордону.
- Ширина і колір меж можуть бути визначені за допомогою звичайних одиниць розміру і кольору CSS.
- У кордонів може бути різний зовнішній вигляд. Найбільш поширені значення **solid**, **double**, **dashed**, **dotted** і **none**.

Приклад використання рамок

- HTML-код

```
<code class="border-solid">2px <br> solid</code>
<code class="border-double">6px <br> double</code>
<code class="border-dashed">8px <br> dashed</code>
<code class="border-dotted">4px <br> dotted</code>
```

- CSS

```
code {
  background: #eaeaed;
  color: #666;
  font: 14px/24px "Source Code Pro", Inconsolata, "Lucida Console", Terminal, "Courier New", Courier;
  display: inline-block;
  height: 70px;
  margin: 0 14px;
  padding-top: 20px;
  text-align: center;
  width: 90px; }
.border-solid {
  border: 2px solid #9799a7; }
.border-double {
  border: 6px double #9799a7; }
.border-dashed {
  border: 8px dashed #9799a7; }
.border-dotted {
  border: 4px dotted #9799a7; }
```

Рамки для окремих сторін елемента

- Як і з властивостями **margin** і **padding** рамки можуть бути розміщені на одній стороні елемента за раз при бажанні. Це вимагає нових властивостей: **border-top**, **border-right**, **border-bottom** і **border-left**. При бажанні можна зробити так, щоб кордон з'являлася тільки внизу елемента:

```
div {
  border-bottom: 6px solid #949599;
}
```

Крім того, стилями для окремих сторін можна управляти на ще більш тонкому рівні. Наприклад, якщо ми хочемо змінити тільки ширину нижньої межі, то можемо використовувати наступний код:

```
div {
  border-bottom-width: 12px;
}
```

Радіус рамки (**border-radius**)

- Властивості **border-radius** дозволяє нам закругляти кути елемента.
- Властивість **border-radius** приймає одиниці розміру, в тому числі відсотки і пікселі, які визначають радіус заокруглення кутів елемента
- Єдине значення заокруглює всі чотири кути елемента в рівній мірі; два значення заокруглюють лівий верхній / правий нижній і правий верхній / лівий нижній кути в такому порядку; чотири значення заокруглюють лівий верхній, правий верхній, правий нижній і лівий нижній кути в такому порядку.
- При розгляді порядку, коли кілька значень застосовуються до властивості **border-radius**, пам'ятайте, що вони йдуть за годинниковою стрілкою, починаючи з лівого верхнього кута елемента.

```
div {
  border-radius: 5px;
}
```

Приклад використання рамок с заокругленими краями

- HTML-код

```
<code class="border-rounded">5px</code>

<code class="border-circle">50%</code>

<code class="border-football">15px 75px</code>
```

- CSS

```
code {
  background: #eaeaed;
  color: #666;
  font: 14px/24px "Source Code Pro", Inconsolata, "Lucida Console", Terminal, "Courier New", Courier;
  display: inline-block;
  height: 90px;
  line-height: 90px;
  margin: 0 14px;
  text-align: center;
  width: 90px; }
.border-rounded {
  border-radius: 5px; }
.border-circle {
  border-radius: 50%; }
.border-football {
  border-radius: 15px 75px; }
```

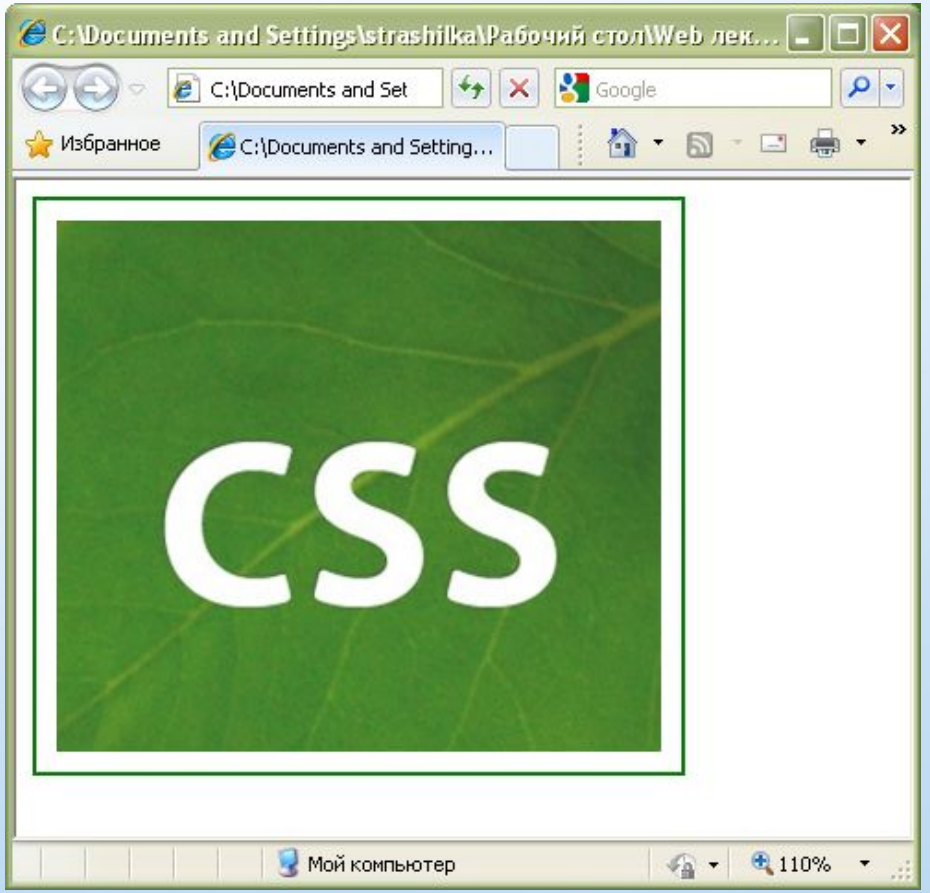
Заокруглення для окремих сторін елемента

- Властивість **border-radius** також може бути розбита на ряд властивостей, які дозволяють нам змінити радіуси окремих кутів елемента. Ці властивості починається з **border**, далі показують положення кута по вертикалі (**top** або **bottom**) і горизонталі (**left** або **right**) і завершуються **radius**.
- Наприклад, для зміни правого верхнього кута може бути використано властивість **border-top-right-radius**.

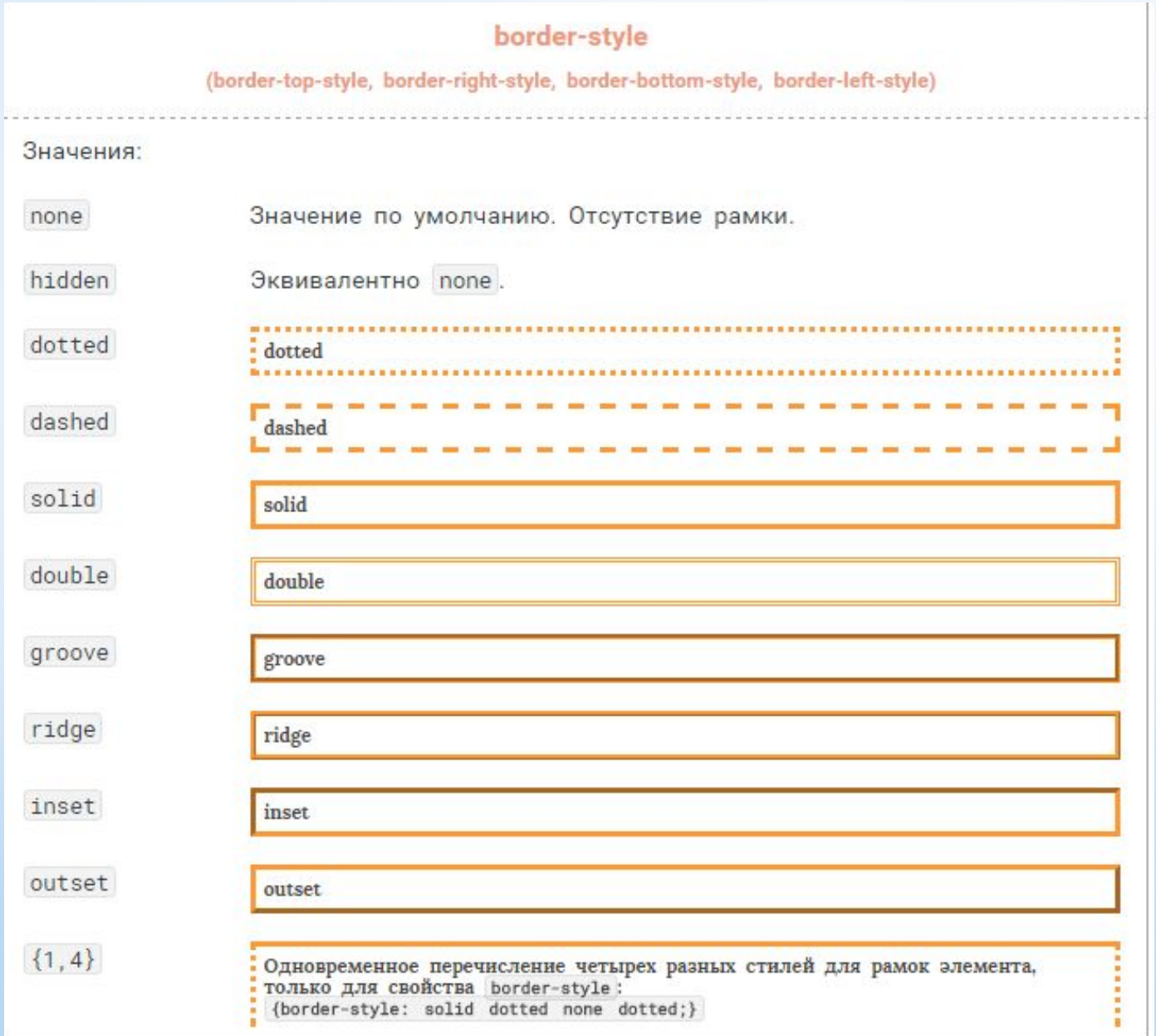
```
div {
  border-top-right-radius: 5px;
}
```

Приклад

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
 <head>
  <style type="text/css">
    img {
      padding: 10px;
      border: 2px solid green;
    }
  </style>
 </head>
 <body>
   <img src="ce.3847b8bfff2ccf223c3c5c2010dd3ace.css2,j.jpg" />
 </body>
</html>
```



- **Стилі рамок**


border-style
(border-top-style, border-right-style, border-bottom-style, border-left-style)
Значения:
none
Значение по умолчанию. Отсутствие рамки.
hidden
Эквивалентно none.
dotted
dotted
dashed
dashed
solid
solid
double
double
groove
groove
ridge
ridge
inset
inset
outset
outset
{1,4}
Одновременное перечисление четырех разных стилей для рамок элемента, только для свойства border-style:
{border-style: solid dotted none dotted;}

- **Колір рамки**

border-color

(border-top-color, border-right-color, border-bottom-color, border-left-color)

Значения:

`transparent`	Устанавливает прозрачный цвет для рамки. При этом ширина рамки остается. Можно использовать для смены цвета рамки при наведении курсора мыши на элемент, чтобы избежать смещение элемента.
`color`	Цвет рамок задается при помощи свойства color элемента.

```
{border-color: #cacd58;}
```

`{1,4}`	Одновременное перечисление четырех разных цветов для рамок элемента, только для свойства `border-color`: `{border-color: #cacd58 #5faf8a #b9cea5 #aab238;}`

- **Ширина рамки**

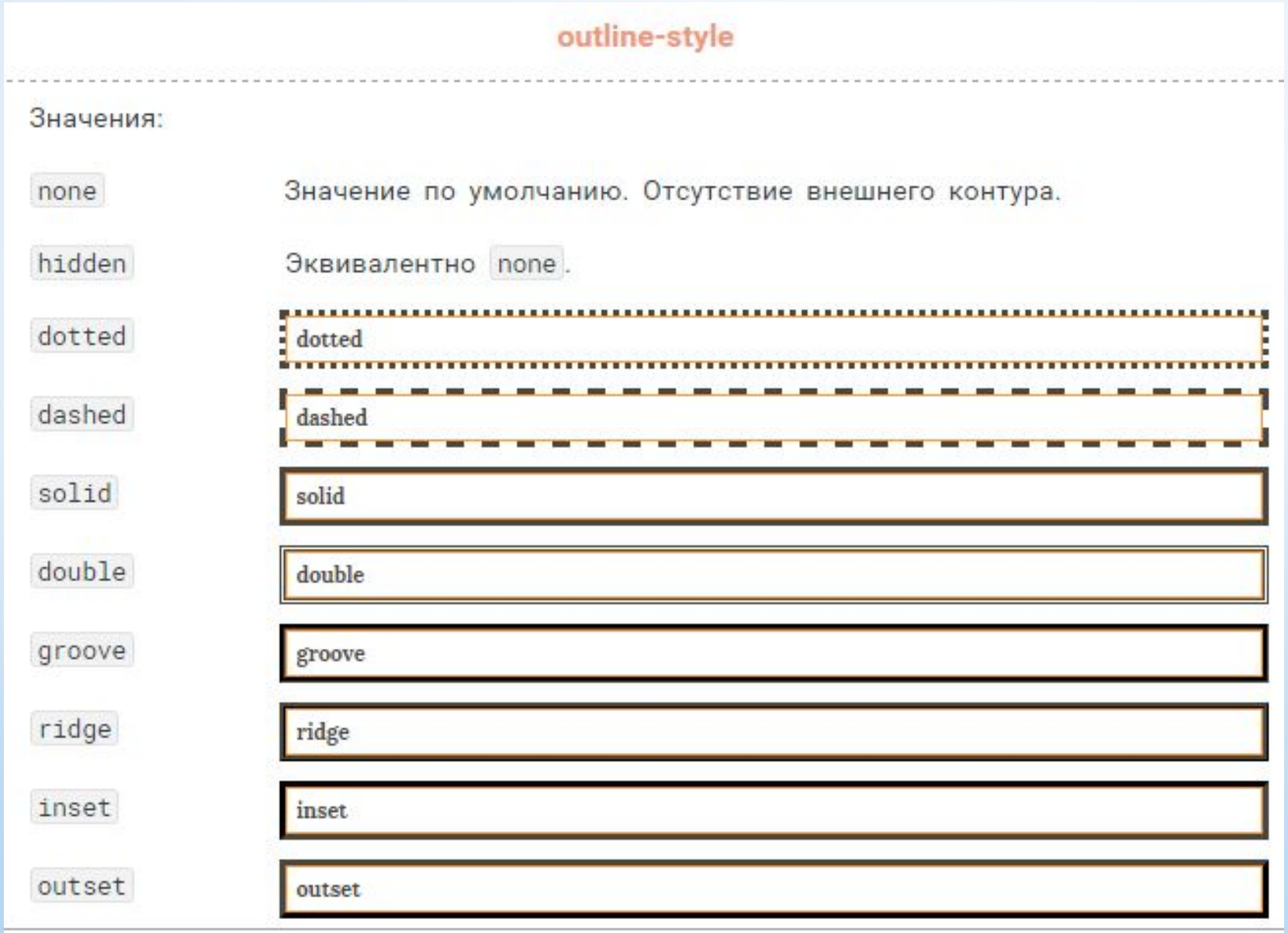
border-width

(border-top-width, border-right-width, border-bottom-width, border-left-width)

Значения:

`thin` / `medium` / `thick`	Ключевые слова, устанавливают ширину рамки относительно друг друга. Первое значение уже, чем второе, второе — тоньше третьего.
`width (px, em, %)`	{border-width: 5px;}
`{1,4}`	Возможность одновременного задания четырех разных ширин для рамок элемента, только для свойства `border-width`: `{border-width: 5px 10px 15px 3px;}`

- **Зовнішній контур рамки**



- **Колір зовнішнього контуру**

outline-color

Значения:

`invert`	Цвет линии (обычно черный) устанавливается автоматически, создавая контраст с основным содержимым.
`color`	Цвет внешней обводки задается при помощи свойства color элемента.

- **Товщина зовнішнього контуру**

outline-width

Значения:

`thin / medium / thick`	Ключевые слова, устанавливают толщину внешнего контура относительно друг друга. Первое значение уже, чем второе, второе — тоньше третьего.
`width (px, em, %)`	{outline-style: dotted; outline-width: 5px;}

3. CSS таблиці

Межі таблиці

- Спочатку таблиця і комірки всередині неї не мають видимих меж.
- Межі таблиці оформляються властивістю border.

```
table {
border-collapse: collapse; /*убираем пустые промежутки между
ячейками*/
border: 1px solid grey; /*устанавливаем для таблицы внешнюю границу
серого цвета толщиной 1px*/
}
th {
border: 1px solid grey;
}
td {
border: 1px solid grey;
}
```

- Так як товщина рамок сусідніх комірок при завданні їх способом **border: 1px solid grey;** НЕ подвоюється, тому, щоб задати рамку для всієї таблиці, досить буде вказати їх для **th, td {border: 1px solid grey;}**.
- Щоб виділити зовнішню рамку таблиці, потрібно встановити, наприклад, **table {border: 3px solid grey;}.**
- Також можна частково задавати рамку як таблиці, так і для комірок, наприклад:

```
table {
border-top: 3px solid grey; /*устанавливаем для таблицы внешнюю
границу серого цвета толщиной 3px*/
}
td {
border-bottom: 1px solid grey;
}
```

Ширина і висота таблиці

- Ширину таблиці і її стовпців можна задати за допомогою властивості **width**.
- Якщо для таблиці задана ширина **table {width: 100%;}**, то вона буде дорівнює ширині блоку-контейнера, а ширина стовпців встановиться відповідно до ширини вмісту комірок.
- Найчастіше ширину таблиці і стовпців задають в px або %.
- Щоб відокремити вміст комірок від рамок, потрібно додати для елементів **<td>** і **<th>** відповідні поля - **padding**.
- Висота таблиці зазвичай не задається. Висотою рядків таблиці можна управляти, додавши верхній і нижній [illegible] допомогою властивості **height**.

```
table {width: 600px;}
th {width: 20%;}
td:first-child {width: 30%;}
th, td {padding: 10px 15px;}
th, td {height: 30px;}
```

Стовпчики таблиці

- Не дивлячись на те, що модель таблиць CSS орієнтована в основному на рядки, проте стовпці грають важливу роль в макеті. Форматування стовпців таблиці можливо наступними способами:
- за допомогою тега **<col>** можна задати фон для будь-якої кількості стовпців,
- за допомогою конструкції **table td: first-child, table td: last-child** можна задати стилі для першого або останнього стовпчика таблиці (за винятком першої комірки заголовка таблиці),
- за допомогою конструкції **table td: nth-child** (правило відбору стовпців) можна задати стилі для будь-яких стовпців таблиці.

Основний заголовок таблиці

- Основний заголовок **<caption>** описує суть вмісту таблиці.
- За допомогою властивості **caption-side** його можна помістити перед таблицею або під нею.
- Для горизонтального вирівнювання тексту заголовка застосовується властивість **text-align**. Успадковується.

caption-side	
Значения:	
top	Заголовок таблицы располагается над таблицей. Значение по умолчанию.
bottom	Располагает заголовок под таблицей.

Рамка комірок таблиці

- За замовчуванням діє модель окремих рамок, яку можна поміняти на загальну модель рамок. Успадковується

.

border-collapse	
Значения:	
collapse	Рамки ячеек будут сливаться.
separate	Рамки ячеек будут располагаться раздельно.

From the People of Japan

UNDP Ukraine

«Frontend-developer: обучающая программа для переселенцев» реализуется в рамках Проекта Програмы развития ООН «Быстрое реагирование на социальные и экономические проблемы внутренне перемещённых лиц в Украине» при финансовой поддержке Правительства Японии.

Приклад

```
table {
border-collapse: collapse;
}
table {
border-collapse: separate;
}
```

Company	Q1	Q2	Q3
Microsoft	20.3	30.5	23.5
Google	50.2	40.63	45.23
Apple	25.4	30.2	33.3
IBM	20.4	15.6	22.3

Company	Q1	Q2	Q3
Microsoft	20.3	30.5	23.5
Google	50.2	40.63	45.23
Apple	25.4	30.2	33.3
IBM	20.4	15.6	22.3

Проміжок між рамками

- Властивість **border-spacing** дозволяє задати відстань між рамками елементів таблиці.
- Властивість застосовується до таблиці в цілому. Проміжок між рамками комірок постійний незалежно від ширини рамок самих коірок. Успадковується.
- **border-spacing** дозволяє розділити осередки проміжком як по вертикалі, так і по горизонталі.
- Якщо задані дві довжини, то перша завжди визначає горизонтальний проміжок, а друга - вертикальний.

4. CSS Посилання

Псевдокласи станів гіпертекстових посилань

- Більшість браузерів виділяють чотири основні стани гіперпосилань, кожному з яких відповідає свій **псевдоклас селектора**:

✔ Невідвідана — **a:link**

✔ Відвідіна — по який вже здійснили перехід — **a:visited**

✔ Ненатиснута — над якою знаходиться вказівник миші — **a:hover**

✔ Натиснута — яка утримується мишкою — **a:active**

Приклад

```
a:link {
   color: #497DDD;
   border-bottom: 1px dashed;
}
a:visited {
   color: #EF7D55;
}
a:hover {
   color: #154088;
   border-bottom: .07em solid;
}
a:active {
   color: #497DDD;
   border-bottom: 1px dashed;
}
```

Стилізація посилань

- Для стилізації окремих посилань потрібно задати їм стильовий клас, після чого можна буде міняти зовнішній вигляд обраних посилань:

```
<a href="http://anysite.ru" class="global">какой-то текст</a>
```

- Прибираємо підкреслення:

```
a {
   text-decoration: none;
}
```

Стилізація посилань

- Додавання підкреслення тільки при наведенні на посилання :

```
a {
  text-decoration: none;
}
a:hover {
  text-decoration: underline;
}
```

- Зовнішній вигляд посилання:

```
a {
  text-decoration: none;
  border-bottom: 2px dashed DarkOrchid;
  padding-bottom: 3px;
}
```

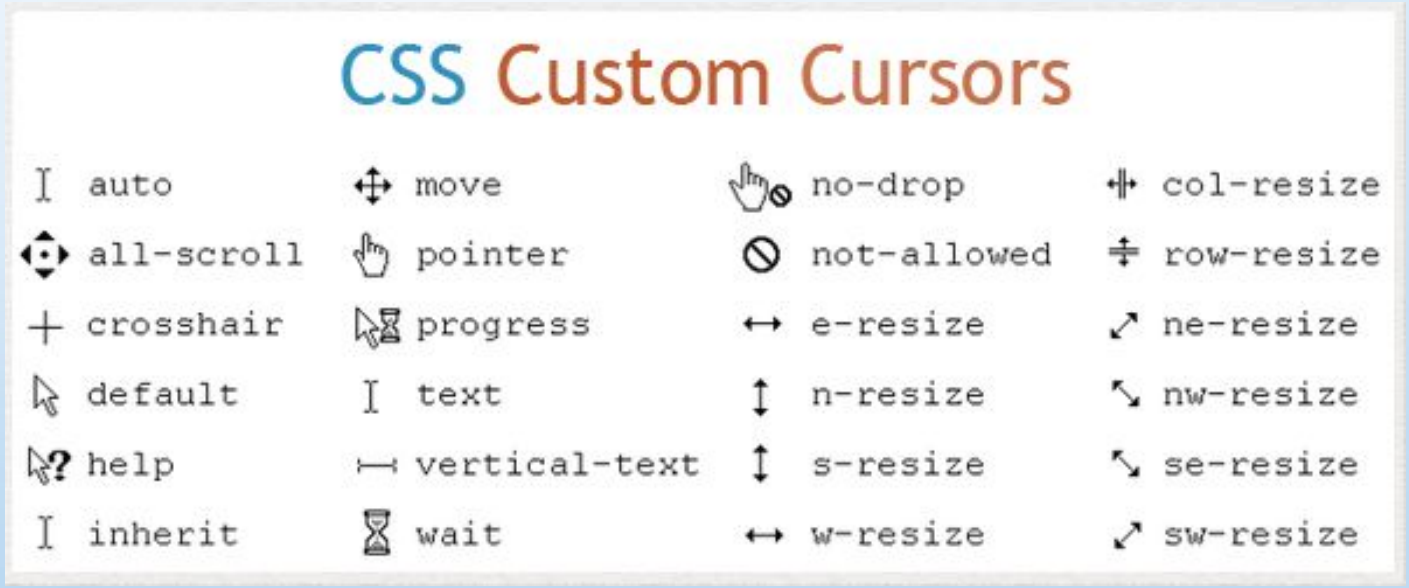
Стилізація посилань

- Можна перетворити зовнішній вигляд посилання, додавши в якості нижньої межі фонове зображення:

```
a {
  text-decoration: none;
  background: url(images/underline.png) repeat-x left bottom;
  padding-bottom: 3px;
}
```

Зовнішній вид курсора миші

- Курсор миші може мати різний вигляд, також можна встановити користувацьке зображення в якості курсора. Наведіть над елементами таблиці нижче, щоб побачити, як виглядає курсор для кожного встановленого значення. Значення за замовчуванням **cursor: pointer;**.



Дякую за увагу

Лектор:

к.е.н., доцент кафедри інформаційного менеджменту

Корзаченко Ольга Володимирівна

Самостійно! Основні типи селекторів CSS

Селектор атрибута

- **Селекторы атрибутов** позволяют форматировать элементы на основе выборки любых содержащихся в них атрибутов или значений атрибутов, варианты:
- **[атрибут]** – выбирает все элементы, для которых задан указанный атрибут.
- **img[alt]** – выбирает все картинки, содержащие атрибут alt.
- **img[title="flower"]** – выбирает все картинки, название которых содержит слово flower.
- **a[href^="http://"]** – выбирает все ссылки, начинающиеся на http://.
- **img[src$=".png"]** – выбирает все картинки, название которых заканчивается на .png.
- **a[href*="ru"]** – выбирает все ссылки, название которых содержит слово ru.
- **input[type="text"]** – выбирает только текстовые поля формы.
- **article[class~="feature"]** – выбирает статьи по частичному значению атрибута, т.е. статьи, название класса которых содержит данное слово.
- **article[id|="feature"]** – выбирает элемент, атрибут которого эквивалентен feature или начинается на feature.

Селектор псевдокласса

- **Псевдоклассы** – это классы, фактически не прикрепленные к тегам html-кода. Они вызывают CSS-правила при совершении того или иного события или подчиняющиеся тому или иному правилу:
- **a:link** – ссылается на непосещенную ссылку.
- **a:visited** – ссылается на уже посещенную ссылку.
- **a:hover** – ссылается на любой элемент, по которому проводят курсором мыши.
- **a:focus** – ссылается на любой элемент, над которым находится курсор мыши.
- **a:active** – ссылается на элемент, который был активизирован пользователем.
- **:valid** – выберет поля формы, содержимое которых прошло проверку в браузере на соответствие указанному типу.
- **:invalid** – выберет поля формы, содержимое которых не соответствует

Селектор атрибута

- **:enabled** – выберет все доступные (активные) поля форм.
- **:disabled** – выберет заблокированные поля форм, т.е., находящиеся в неактивном состоянии.
- **:in-range** – выберет поля формы, значения которых находятся в заданном диапазоне.
- **:out-of-range** – выберет поля формы, значения которых не входят в установленный диапазон.
- **:lang()** – выбирает абзацы на указанном языке.
- **:not(селектор)** – выберет элементы, которые не содержат указанный селектор, например, класс, идентификатор или селектор элемента :not([type="submit"]).

:target – выбирает элемент с символом #, на который ссылаются в документе.

Структурные псевдоклассы

- Структурные псевдоклассы форматируют дочерние элементы в соответствии с указанным параметром в скобке, варианты:
- **:nth-child(odd)** – выбирает нечетные дочерние элементы.
- **:nth-child(even)** – выбирает четные дочерние элементы.
- **:nth-child(3n)** – выбирает каждый третий элемент среди дочерних.
- **:nth-child(3n+2)** – выбирает каждый третий элемент, начиная со второго дочернего элемента (+2).
- **:nth-child(n+2)** – выбирает все элементы, начиная со второго.

Структурные псевдоклассы

- **:nth-last-child()**– в списке дочерних элементов выбирает элемент с указанным местоположением, аналогично с **:nth-child()**, но начиная с последнего, в обратную сторону.
- **:first-child** – позволяет оформить только самый первый дочерний элемент тега.
- **:last-child** – позволяет форматировать последний дочерний элемент тега. **:only-child** – выбирает элемент, являющийся единственным дочерним элементом.
- **:empty** – выбирает элементы, у которых нет дочерних элементов.

:root – выбирает элемент, являющийся корневым в документе (элемент html)

Структурные псевдоклассы типа

- Позволяют указать на конкретный тип дочернего тега:
- **:nth-of-type()** – выбирает элементы по аналогии с :nth-child(), при этом берет во внимание только тип элемента.
- **:first-of-type** – позволяет выбрать первый дочерний элемент.
- **:last-of-type** – выбирает последний тег конкретного типа.
- **:nth-last-of-type()** – выбирает элемент заданного типа в списке элементов в соответствии с указанным местоположением, начиная с конца.
- **:only-of-type** – выбирает единственный элемент указанного типа среди дочерних элементов родительского элемента.

Селекторы псевдоэлементов

- **Псевдоэлементы** используются для добавления содержимого, которое генерируется с помощью свойства content, и для изменения внешнего вида части элемента:
- **:first-letter** – выбирает первую букву каждого абзаца, применяется только к блочным элементам.
- **:first-line** – выбирает первую строку текста элемента, применяется только к блочным элементам.
- **:before** – вставляет генерируемое содержимое перед элементом.
- **:after** – добавляет генерируемое содержимое после элемента.