Web-технології та web-дизайн. CSS

2024/2025

Самостійна робота 2
до лекції №4

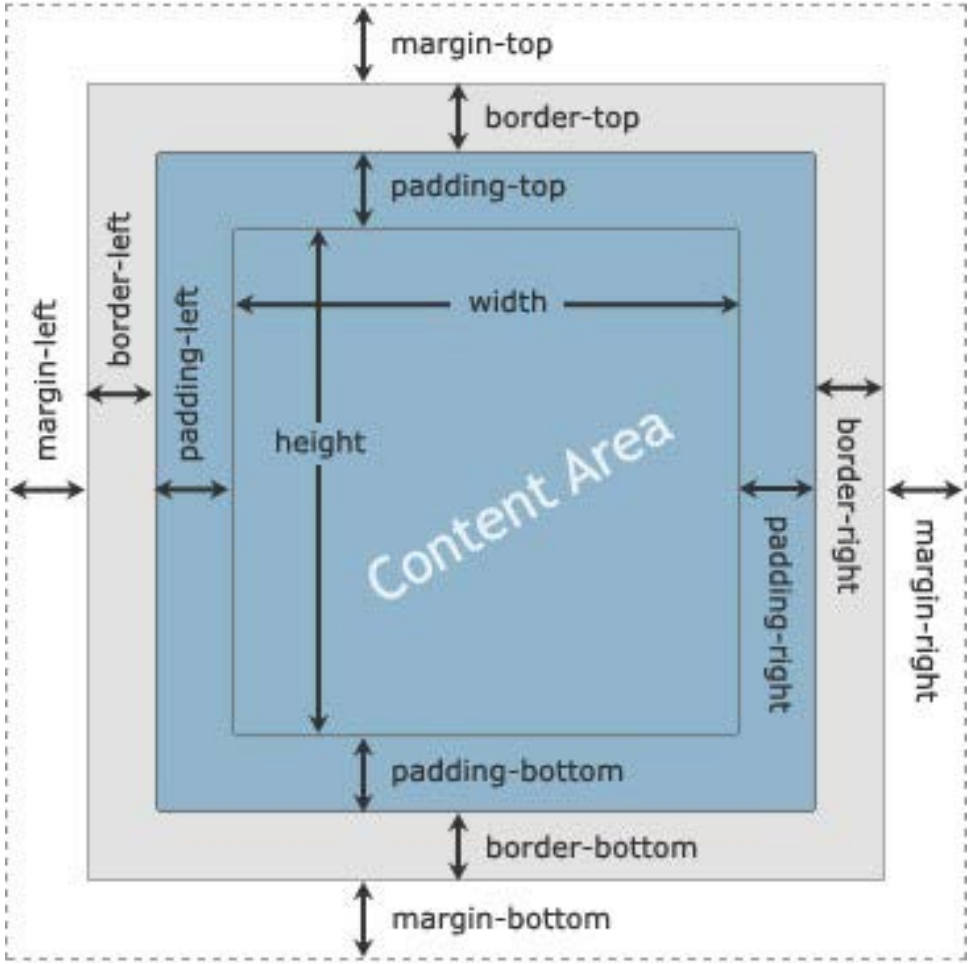
Лекції: ст. викладач каф. Штучного інтелекту
Гриньова Олена Євгенівна
olena.hrynova@nure.ua

Зміст

Властивості боксової моделі CSS
Властивості блокового елемента: висота, ширина, позиція
Властивість z-index, відступи, рамки, центрування
Рядкові елементи
Рядково-блокові елементи
CSS властивість display

Властивості Box моделі CSS

Box Model



У кожного боксу є 4 області:
margin (зовнішні відступи),
border (рамка),
padding (внутрішні відступи),
content (контент або зміст).

Стандартна Box модель

Скільки ж в підсумку місця буде займати бокс?

Область, яка займається **блоковим** боксом, складається з його ширини та висоти змісту, внутрішніх і зовнішніх відступів, ширини рамок.

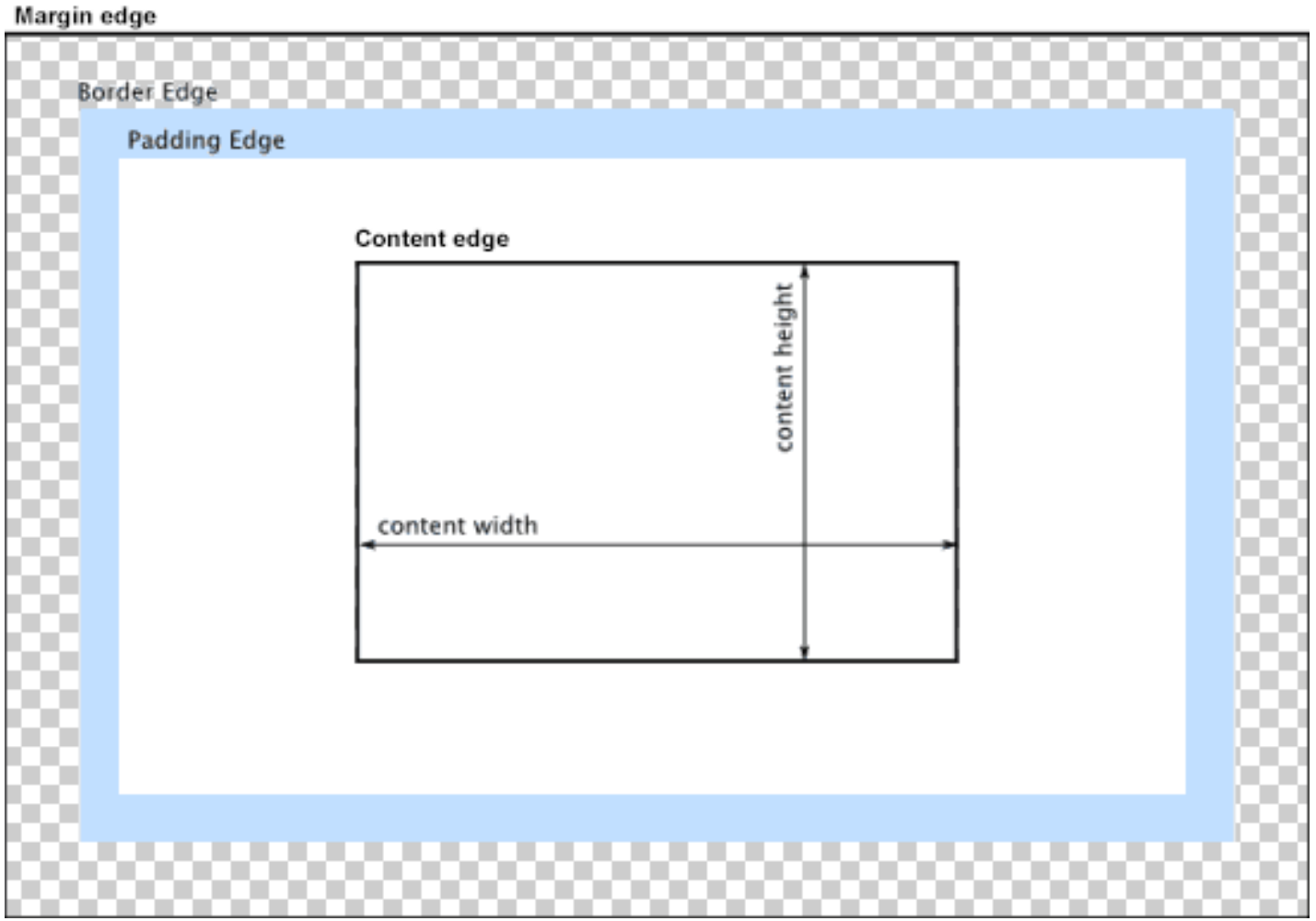
Для **рядкових** боксів існують свої особливості.

Box модель, content area

Внутрішня область елемента (**content area**) містить текст та інші елементи, які розташовані всередині (контент або зміст). У неї часто буває фон, колір або зображення (в такому порядку: фоновий колір ховається під непрозорим зображенням), і вона знаходиться всередині **content edge**.

Її розміри - ширина контенту (**content width** або **content-box width**), і висота контенту **(content height** або **content-box height**). Іноді ще кажуть «**внутрішня ширина/висота елемента**».
За замовчуванням, якщо CSS-властивість **box-sizing** не задана, розмір внутрішньої області зі змістом задається властивостями **width, min-width, max-width, height, min-height** і **max-height.** Якщо ж властивість **box-sizing** задано, то вона визначає, для якої області вказані розміри.

Box модель


Margin edge
Border Edge
Padding Edge
Content edge
content height
content width

Блокова модель, padding area

Поля елемента (**padding area**) — це порожня область, яка оточує контент. Вона може бути залита якимось кольором, покрита фоновою картинкою, а її межі називаються краями внутрішніх відступів (**padding edge**).

Розміри внутрішніх відступів задаються окремо з різних сторін властивостями **padding-top, padding-right, padding-bottom, padding-left** або загальної властивістю **padding**.

Блокова модель, border area

Область рамки (**border area**) оточує поля елемента, а її межа називається край рамки (**border edge**). Ширина рамки задається окремою властивістю **border-width** або в составі властивості **border**. Розміри елемента з урахуванням внутрішніх відступів і рамки іноді називають **зовнішньою шириною/висотою елемента.**

Блокова модель, margin area

Зовнішні відступи (**margin area**) додають порожній простір навколо елемента та визначають відстань до сусідніх елементів.
Величина зовнішніх відступів задається окремо в різних напрямах властивостями **margin-top, margin-right, margin-bottom, margin-left** або спільною властивістю **margin**.

Зовнішні відступи двох сусідніх елементів, розташованих один над одним або вкладений один в інший, можуть накладатись. Це називається **схлопуванням** меж (**margin collapsing**).

Схлопуються тільки вертикальні відступи.

Висота блокового елемента

height: **n**px | **n**%

встановлює висоту блокового елемента, висота не включає в себе товщину границь навколо елемента, значення внутрішніх і зовнішніх відступів.

div {height: 150px}

Висота за замовчуванням блокових боксів залежить від їх змісту. Якщо задати блоковому боксу ширину та висоту так, що контент не буде в ньому вміщатись, то він як би «випаде» з нього.

.h {height: 150px}

…

Повернути значення **за замовчуванням** можна за допомогою спеціального значення **auto**:

height: auto;

Ширина блочного елемента

width:**n**px | **n**%

встановлює ширину блочного елемента, ширина не включає в себе товщину границь навколо елемента, значення внутрішніх і зовнішніх відступів.

За замовчуванням блокові бокси займають усю доступну ширину, яка **дорівнює ширині базового контейнера або вікна браузера.**

```
div {width: 100px}
.w {width: 100px}
```

```
<div class=w>…</div>
```

Повернути значення **за замовчуванням** можна за допомогою спеціального значення **auto**:

```
width: auto;
```

Позиція

position: static | absolute | fixed | relative

position визначає спосіб позиціонування блокового елемента відносно вікна браузера або інших елементів.

```
div {position: absolute; top: 10%; left: 10%}
.pos {position: absolute; top: 10%; left: 10%}
```

```
<div class= pos >…</div>
```

Позиція

за замовчуванням

position: static

переміщування відносно його поточної позиції

position: relative

переміщування відносно першого позиціонованого базового елемента (relative, absolute або fixed)

position: absolute

працює як абсолютне, точкою відліку є вікно перегляду

position: fixed

«Координати»

top, left, right, bottom :**n**px | **m**%

Для розташованого елемента визначає відстань від лівого краю батьківського елемента, не включаючи зовнішні і внутрішні відступи, ширину рамки, до лівого краю дочірнього елемента (вкладеного).

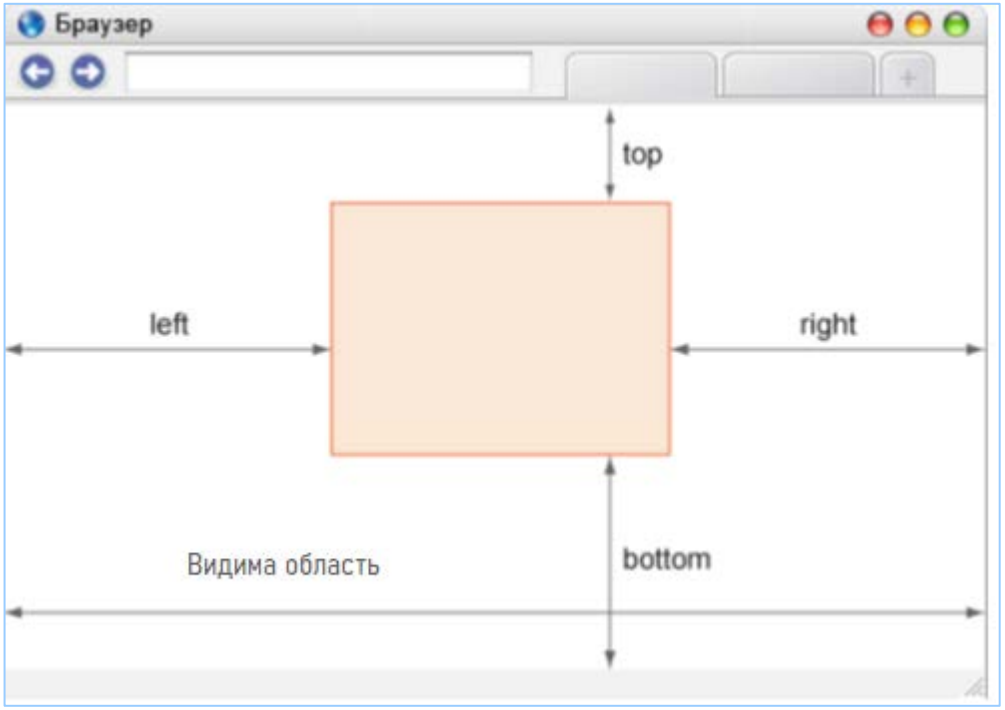
Властивість не працює для елементів, у яких position не встановлений або встановлений явно position: static (за замовчуванням)

```
div {top: 10%; left: 10%}
.pos {top: 10%; left: 10%}
```

…

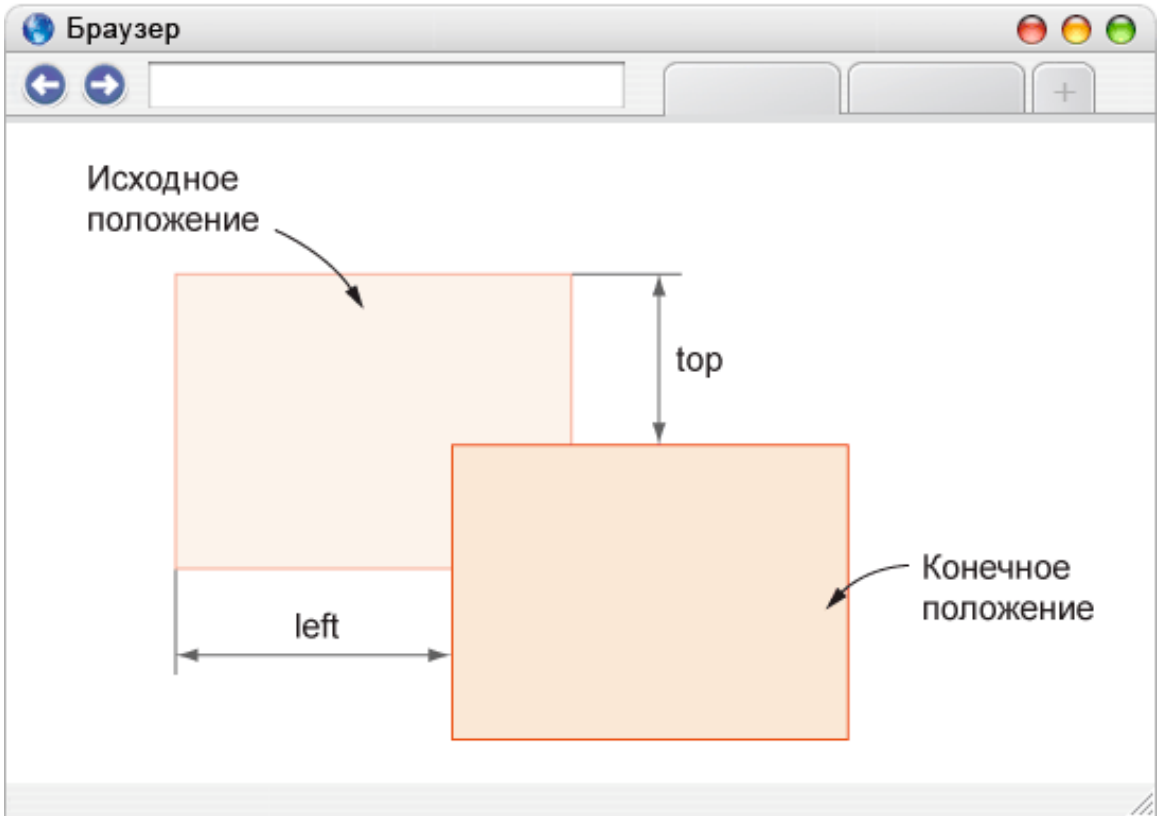
«Координати» + absolute

Якщо position: absolute, в якості батьківського елемента виступає вікно браузера та положення елемента визначається від його лівого краю. left + right або left + width



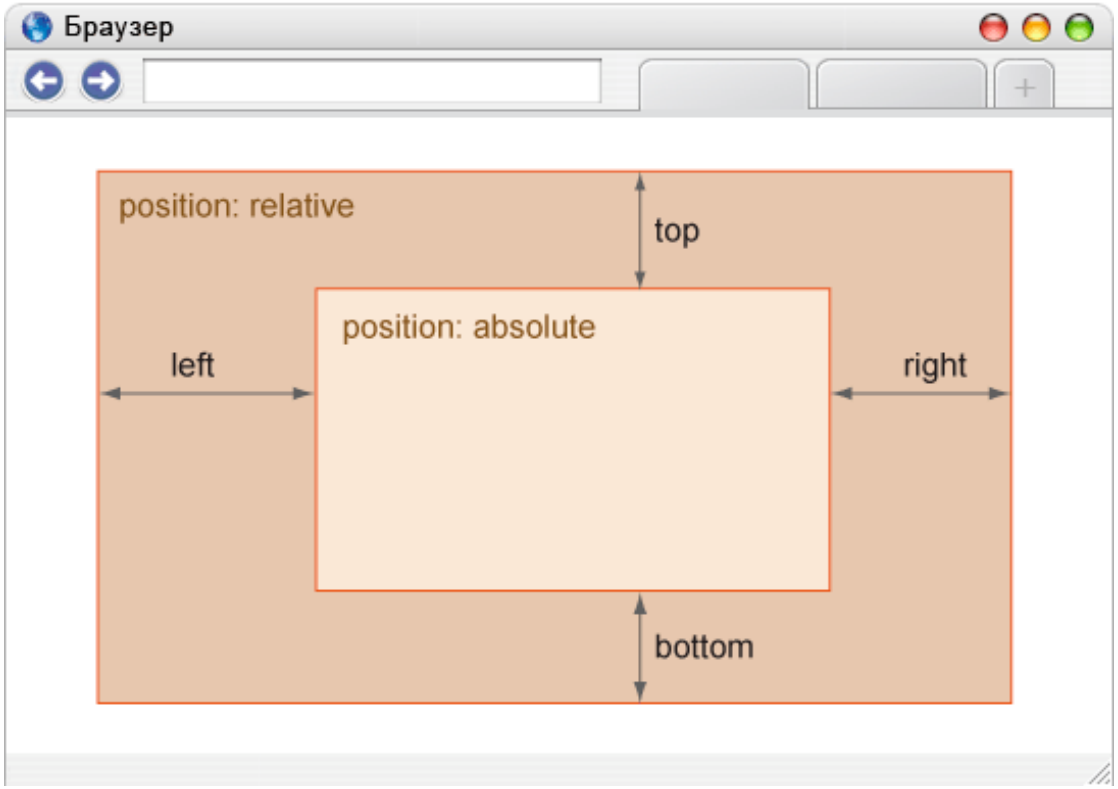
«Координати» + relative

Якщо position: relative, в якості батьківського виступає початкове положення елемента.



«Координати» relative + absolute

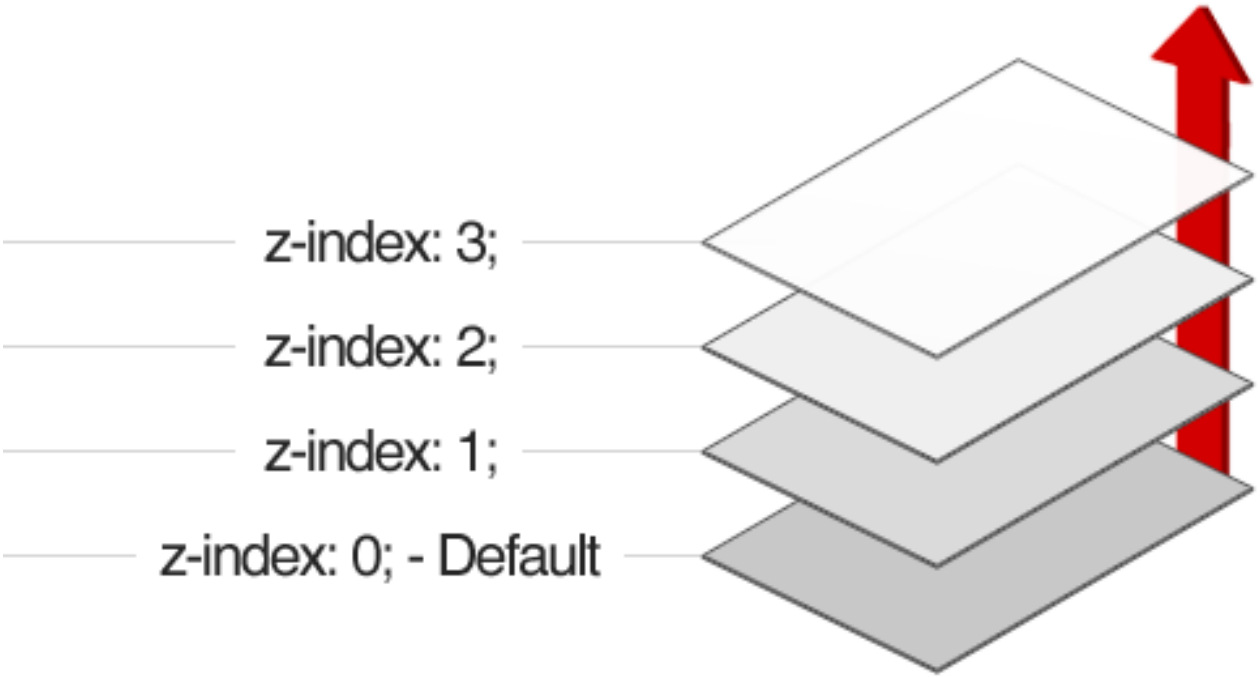
Відлік координат ведеться від внутрішнього краю межі, значення padding не враховуються.



Властивість z-index

z-index: **n**

Для позиціонованого елемента визначає положення блокового елемента "в глибині"



Властивість z-index

Будь-які позиціоновані елементи на веб-сторінці можуть накладатися один на одного в певному порядку, імітуючи тим самим третій вимір, перпендикулярний екрану. Кожен елемент може бути як під, так і над іншими об'єктами на веб-сторінці, їх розміщенням по **z-вості** керує **z-index**. Ця властивість працює лише для елементів, у яких значення **position** встановлено як **absolute, fixed** або **relative**.

z-index: число | auto | inherit

В якості значення використовуються цілі числа (додатні, від'ємні та нульові). **Чим вище значення, тим вище знаходиться елемент в порівнянні з тими елементами, у яких воно менше.**

При рівному значенні **z-index**, на передньому плані знаходиться той елемент, який в HTML-коді описаний нижче.

Без z-index

У приведеному нижче прикладі елементи з 1 по 4 позиціонуються. Елемент №5 статичний, тому він намальований під чотирма іншими елементами, хоча з'являється пізніше в розмітці HTML.

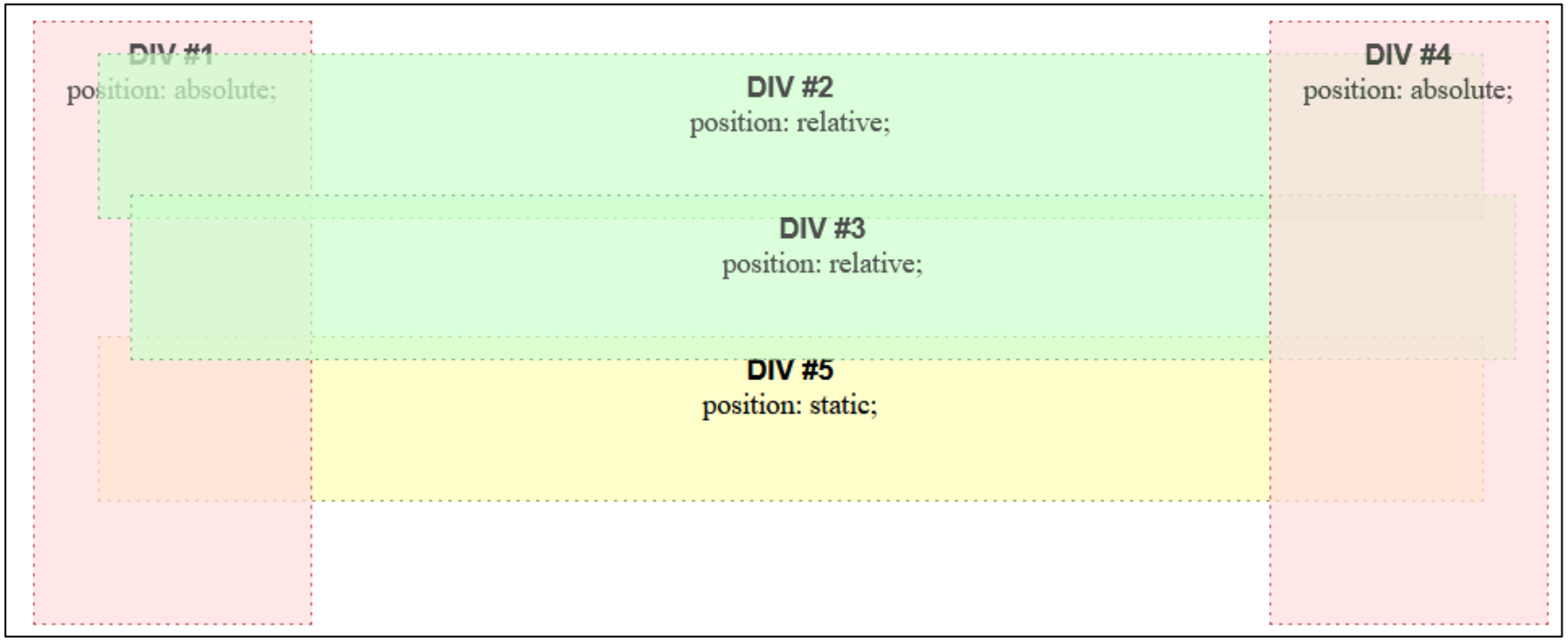
HTML

```
<div id="abs1" class="absolute">
  <b>DIV #1</b><br />position: absolute;</div>
<div id="rel1" class="relative">
  <b>DIV #2</b><br />position: relative;</div>
<div id="rel2" class="relative">
  <b>DIV #3</b><br />position: relative;</div>
<div id="abs2" class="absolute">
  <b>DIV #4</b><br />position: absolute;</div>
<div id="sta1" class="static">
  <b>DIV #5</b><br />position: static;</div>
```

CSS

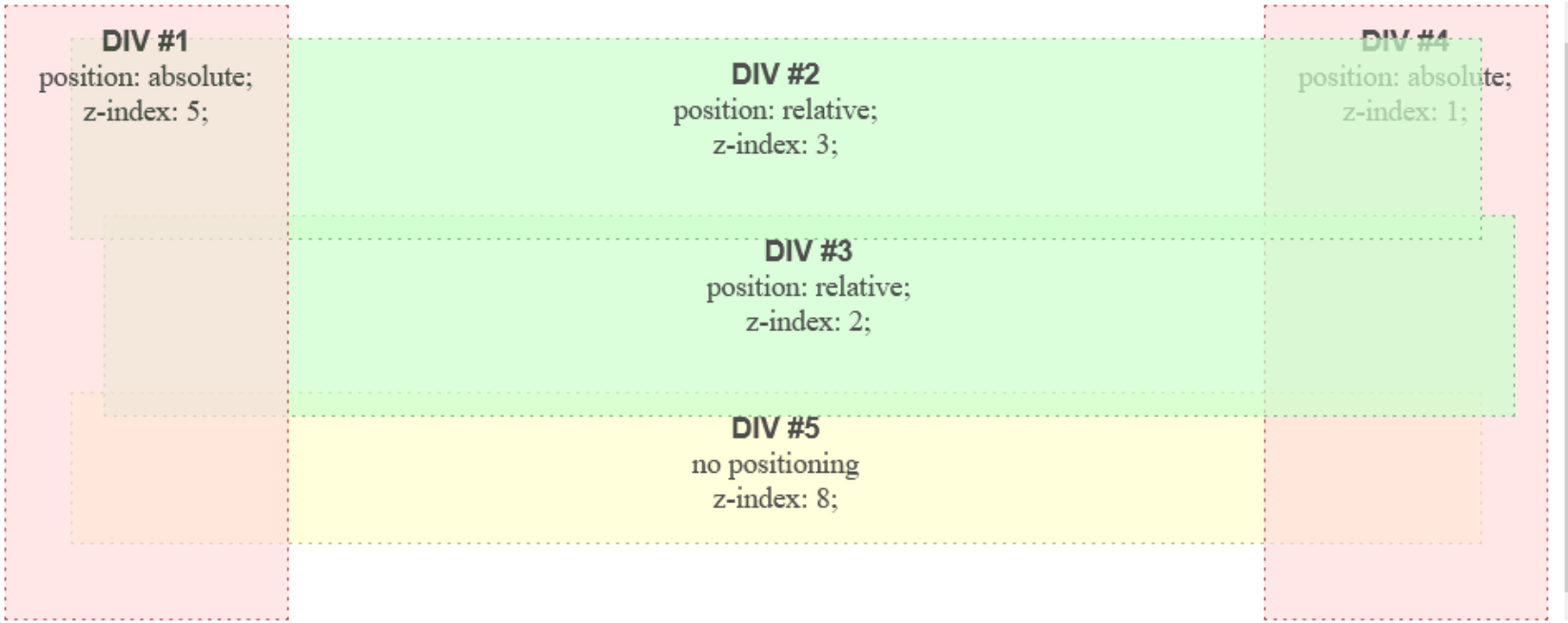
```
div {  padding: 10px;
  border: 1px dashed;
  text-align: center;}
.static {  position: static;
  height: 80px;
  background-color: #ffc;
  border-color: #996;}
.absolute {  position: absolute;
  width: 150px;
  height: 350px;
  background-color: #fdd;
  border-color: #900;
  opacity: 0.7;}
.relative {  position: relative;
  height: 80px;
  background-color: #cfc;
  border-color: #696;
  opacity: 0.7;}
```

Без z-index


DIV #1
position: absolute;
DIV #2
position: relative;
DIV #4
position: absolute;
DIV #3
position: relative;
DIV #5
position: static;

Властивість z-index

У наступному прикладі порядок укладання шарів змінений за допомогою **z-index**. z-index для елемента №5 не має ніякого ефекту, тому що це **не позиціонований елемент**. Коли для **z-index** властивість **position** не вказана, елементи відображаються на рівні візуалізації **за замовчуванням - 0 (нуль).**



Відступи

Внутрішні відступи, padding

Властивість **padding** встановлює внутрішні відступи блоку — відступи від зовнішньої межі блоку до його контенту. Ці відступи іноді називають «полями».

Внутрішні відступи для різних сторін встановлюються за допомогою властивостей **padding-top, padding-right, padding-bottom** и **padding-left**:

Внутрішні відступи, padding

Однакові відступи з усіх сторін
padding: 10px;
Відступи зверху и знизу 5px, праворуч и ліворуч 10px
padding: 5px 10px;
Відступи зверху 5px, ліворуч и праворуч 10px, знизу 15px.
padding: 5px 10px 15px;
Різні відступи з усіх сторін, по порядку **верхній, правий, нижній, лівий**
padding: 5px 10px 15px 20px;

Для строкових боксів краще не встановлювати вертикальні відступи, так як вони поводяться непередбачувано.

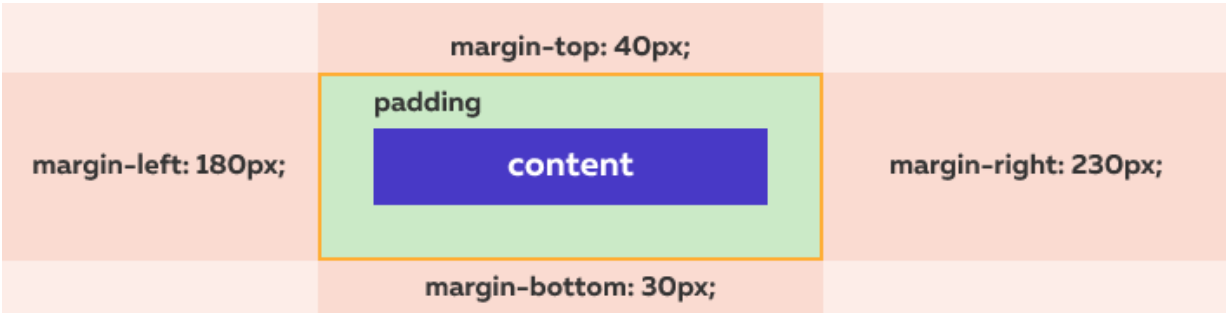
Зовнішні відступи, margin

Властивість **margin** встановлює зовнішні відступи блоку — відступи від зовнішньої межі елемента до границь батьківського елемента або до сусідніх елементів.

Зовнішні відступи для різних сторін встановлюються за допомогою властивостей **margin-top, margin-right, margin-bottom** та **margin-left**.

Скорочена властивість **margin** працює аналогічно властивості padding, тільки встановлює зовнішні відступи, а не внутрішні.

Строкові бокси реагують **тільки на горизонтальні внутрішні відступи.**



«Згортання» зовнішніх відступів, margin collapsing

Властивості блоків **margin-top** и **margin-bottom** іноді комбінуються (згортаються) у єдиний відступ, розмір якого дорівнює найбільшому з комбінованих відступів. Така поведінка відома як **згортання зовнішніх відступів.**

Вертикальний зовнішній відступ між двома сусідніми елементами дорівнює максимальному зовнішньому відступу між ними. Якщо зовнішній відступ одного елемента 20px, а іншого 40px, зовнішній відступ між ними буде 40px.

Горизонтальні зовнішні відступи між елементами просто складаються. Наприклад, горизонтальний зовнішній відступ між двома елементами із зовнішніми відступами 30px становитиме 60px.

Зовнішні відступи плаваючих і абсолютно позиціонованих елементів ніколи не згортаються.

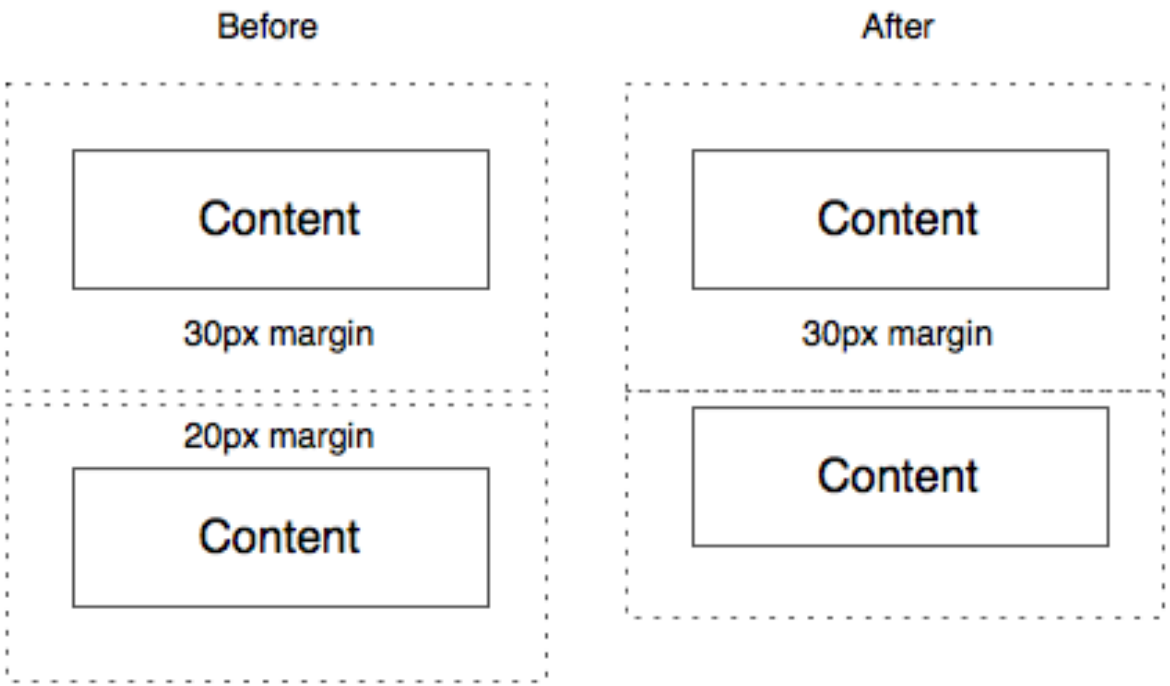
«Згортання» margin

Зовнішній відступ між сусідніми елементами буде 50px.

Випадки «згортання» margin

1. **Сусідні елементи (siblings)**. Згортаються відступи сусідніх елементів (за винятком випадка, коли до останнього елемента застосована властивість **clear**).
2. **Батьківський та перший/останній дочірні елементи.** Якщо відсутні границі (border), внутрішні відступи (padding), строковий вміст (inline/inline-block) або проміжок для відділення margin-top батьківского елемента від margin-top одного або декількох його дочірніх елементів/блоків, то зовнішні відступи згортаються. Згорнуті відступи закінчуються за межами батьківського елемента.
3. **Порожні блоки.** Якщо відсутні границі (border), внутрішні відступи (padding), строковий вміст (inline/inline-block), height або min-height для відділення margin-top верхнього відступу цього блоку від його margin-bottom нижнього відступу, то верхні і нижні зовнішні відступи цього блоку згортаються.

«Згортання» margin



«Випадання» margin

Якщо всередині батьківського блоку розмістити блок і задати йому відступ зверху, то внутрішній блок буде притиснутий до верхнього краю батьківського блоку, а у батьківського елемента з'явиться відступ зверху. Тобто верхній відступ внутрішнього елемента «випадає» з батьківського елемента.

Щоб позбутися від ефекту випадання, можна встановити батьківському елементу внутрішній відступ **padding** зверху (зовнішні та внутрішні відступи завжди складаються), або додати рамку зверху.

Якщо у батьківського елемента також був встановлений зовнішній відступ, то буде вибрано максимальний відступ між власним і «випадним».

Скасування згортання

- Для елементів встановлена властивість **padding;**
- для елементів на стороні згортання встановлена межа;
- на елементах з абсолютним позиціонуванням, тобто тих, у яких **position** встановлено як **absolute**;
- на плаваючих елементах (для них властивість **float** встановлена як **left** або **right**);
- для строкових елементів;
- для **<html>**

Рамки, border

Рамка задається за допомогою властивості **border**, яка складається з трьох компонентів: **ширина рамки**; **стиль рамки**; **колір.**

```
selector {
  border: 5px solid red;
}
```

Це правило встановлює червону суцільну рамку товщиною 5px.

Деякі з найбільш поширених стилів рамок:

- **solid** — суцільна;
- **dashed** — пунктирна;
- **dotted** — крапками.

Задавати рамку можна однією властивістю **border**, а можна и за допомогою окремих властивостей **border-width, border-style, border-color.**

```
selector {border-width: 5px; border-style: solid; border-color: red; }
```

Рамку можна задавати елементам і зі строковим, і з блоковим боксом.

Рамки, border-style

Властивість **border-style** встановлює стиль границі навколо елемента. Допустимо встановлювати індивідуальні стилі для різних сторін елемента.
border-style: [none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset] {1,4} | inherit

none - не відображає межу та її товщина (border-width) встановлюється нульовою.

hidden - має такий самий ефект, що і **none** за винятком використання **border-style** до комірок таблиці, у якої значення властивості **border-collapse** встановлено як **collapse**. У цьому випадку межа навколо комірки взагалі не відображатиметься. **none** має найнижчий пріоритет, **hidden** - найвищий.

inherit – успадковує батьківське значення.

Стиль границі в різних браузерах може трохи відрізнятися при використанні значень **groove, ridge, inset** або **outset**.

Кожна сторона може мати свій власний набір властивостей, які зручно використовувати для скасування одного (**border-right-style**)

Рамки, border-color

Властивість **border-color** встановлює колір межі з різних сторін елемента. Властивість дозволяє встановити колір межі для всіх сторін елемента одночасно або тільки для вказаних.

border-color: [цвет | transparent] {1,4} | inherit

1 - колір межі буде встановлено для всіх сторін елемента.

2 - перше значення встановлює колір верхньої і нижньої межі, друге - лівої і правої.

3 - перше значення встановлює колір верхньої межі, друге – одночасно лівої і правої межі, а третє – нижньої межі.

4 - по черзі встановлює колір верхньої, правої, нижньої та лівої межі.

transparent – встановлює прозорий колір.

inherit – успадковує батьківське значення.

Кожна сторона може мати свій власний набір властивостей, які зручно використовувати для скасування одного (**border-right-color**)

Форматування окремих меж

Ви можете керувати межою з кожного боку елемента окремо за допомогою відповідної властивості: **border-top, border-bottom, border-left** або **border-right**. Вони працюють точно так само, як і стандартне **border**, за винятком, що контролюють межу лише з одного боку стилізованого елемента.

```
border: 2px solid black;
border-bottom: 4px dashed red;
```

Остання явна властивість може перевизначити будь-які подібні властивості, визначені в коді CSS вище. Це приклад того, як працює каскадний механізм CSS.

Центрування

Щоб відцентрувати елемент блоку, потрібно виконати такі дії:

- встановити елементу ширину, меншу за ширину батьківського контейнера.
- встановити для зовнішніх відступів праворуч і ліворуч значення **auto**.
- або просто використовувати **FlexBox.**

```
selector {
  width: 100px;
  margin-left: auto;
  margin-right: auto;
}
```

Box модель и строкові бокси

Особливості поведінки елементів зі строковим боксом в боксовій моделі:

- Не реагують на CSS-властивості **width** і **height**.
- Частково реагують на **margin**, сприймаючи лише горизонтальні відступи.
- Частково реагують на **padding**, сприймаючи лише горизонтальні відступи.
- При встановленні вертикальних **padding** візуально збільшуються, але без збільшення зайнятого простору (інші елементи не відштовхують).
- Сприймають рамки. Аналогічно **padding** рамки зверху і знизу не збільшують зайнятого елементом простору.

Box модель і строкові бокси

CSS-властивість **width** встановлює не загальну ширину блоку, а **лише ширину контенту**. **Загальна ширина блоку** потім складається з трьох компонентів: ширини вмісту, внутрішніх відступів і ширини рамок ліворуч і праворуч.
Поведінка елемента може залежати від того, як саме ви визначаєте його ширину.

Ширина блоку встановлена **явно**, наприклад, width: 100%;.
У цьому випадку ширина вмісту блоку дорівнює ширині батьківського блоку. Якщо до блоку додати внутрішні відступи та рамки, то його загальна ширина стає більшою за батьківську ширину.

Box модель і строкові бокси

За замовчуванням, коли ширина не встановлена, це відповідає значенню width: auto;. У цьому випадку блок займає всю ширину батьківського блоку. Якщо блок має внутрішні відступи або рамки, його ширина вмісту автоматично зменшується, а загальна ширина залишається рівною батьківської ширині.

Крім CSS ширина полів вводу може бути встановленою в значенні атрибута **size**. Ширина width: auto; для полів вводу обчислюється зі значення **size** за замовчуванням і не розтягує поля на всю ширину контейнера.

Строкові бокси, box-sizing.

У **моделі IE** властивість **width** відображала кінцеву ширину блоку, у той час як у **моделі W3C (стандартної) вона** відображала ширину контентної частини блоку (тобто частини блоку, в якій безпосередньо розташовується його вміст). Щоб змусити IE грати за правилами, необхідно було використовувати **хак Тантека Челіка** для Box моделі.
http://tantek.com/CSS/Examples/boxmodelhack.html

CSS-властивість **box-sizing** дає можливість перемикатися між **стандартною блоковою моделлю box-sizing:content-box** і **старою моделлю IE box-sizing: border-box**.

Строкові бокси, box-sizing.

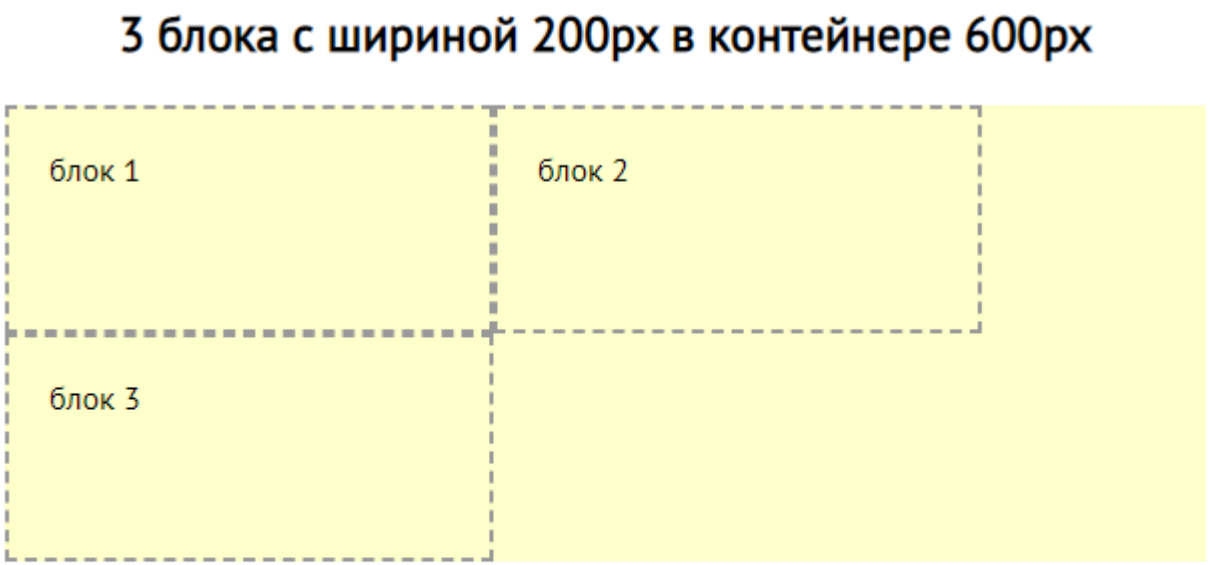
Такі проблеми можна вирішити за допомогою CSS3 властивості **box-sizing**. Одним зі значень цієї властивості є **border-box**, що дозволяє не враховувати відступи і границі в розмірах блоків. Таким чином, вказавши розміри блоку, ви вказуєте їх всьому блоку, а не вмісту.

Властивість **box-sizing** має два значення:

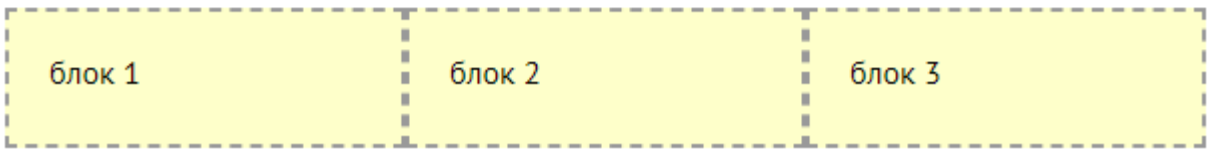
- **content-box** — значення за замовчуванням, відповідає стандартній моделі блоків.
- **border-box** — змінює режим розрахунку ширини елемента на описаний вище: тепер ширина елемента включає і рамку, і внутрішні відступи і, по суті, ширину змісту самого елемента.

Строкові бокси, box-sizing.

content-box



border-box



Керування типом боксу, display

Тип боксу елемента не є чимось вічним і незмінним, його можна змінювати за допомогою CSS. За це відповідає властивість **display**.

З його допомогою, наприклад, можна зробити бокси абзаців і заголовків строковими, а бокси спанів і стронгів - блоковими елементами.

У властивості **display** багато значень.
Властивість display: block; позначає **блоковий** бокс елемента.
Властивість display: inline; — строковий бокс елемента.

Інструкція **display:**inline-block залишить елемент **лінійним**, але він буде сприйматися як блоковий, тому до нього будуть застосовані відступи, поля, межі, ширина і висота.

Властивість display: inline-block

Іноді виникає необхідність розташувати в ряд кілька елементів з заданими розмірами. Елементи зі строковим боксом для цього не підходять, так як не сприймають розмірів. Елементи з блоковим боксом також не підходять, так як до і після них відбувається розрив рядка. Рішення — використовувати елементи з **блоково-строковим боксом**. В HTML немає елементів з блоково-строковим боксом за замовчуванням, однак будь-який елемент можна переключити на цей режим відображення, надавши йому властивість display: inline-block;.

Особливості елементів з блоково-строковим боксом:

їм можна встановлювати розміри, рамки і відступи, як і елементам з блоковим боксом;

- їх ширина за замовчуванням залежить від змісту, а не розтягується по всій ширині контейнера;
- Вони не породжують примусові розриви рядків, тому можуть перебувати на одному рядку, поки вміщуються в батьківський контейнер.
- елементи в одному рядку вирівнюються вертикально, як елементи зі строковим боксом.

Питання для самоконтролю

Властивості блокового елемента: висота, ширина, позиція.
Властивість z-індекс, відступ, рамки, центрування.
Строкові бокси.
Строково-блокові бокси.
Керування типом боксу.

Дякую за увагу!