**ЛАБОРАТОРНА РОБОТА №5**

**CSS властивості оформлення блоків (border, box-shadow).**

**«Плаваюча» модель для розміщення зображень й інших елементів**

# Теоретичний матеріал

Перелік основних CSS-властивостей:

## 1. Границі border

Параметр дозволяє одночасно встановити стиль (border-style), колір (border-color) і товщину (border-width) границі навколо елемента. Значення можуть йти в будь-якому порядку, розділяючись пробілом, браузер сам визначить, яке з них відповідає потрібному атрибуту.

Синтаксис:



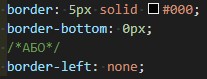
Властивість border не застосовна до рядкових тегів, таких як span, проте застосовна до тегу img.

Властивість border впливає на вміст блоку, а точніше, блок зміщується (відступає) всередину на товщину границі. Тобто, якщо товщина блоку 5px, то вмістиме блоку зміститься на 5px зліва та на 5px справа, проте при цьому, загальна ширина блоку не зміниться.

Якщо ми хочемо встановити границю для обраної сторони блоку, то тоді потрібно використовувати запис границі для кожної сторони окремо:  border-bottom – нижня границя.

* border-top – верхня границя.
* border-left – ліва границя.
* border-right – права границя.

Якщо нам непотрібна тільки одна сторона границі, а всі решта повинні бути задані, то тоді використовують запис:



Однак, запис відсутності однієї із сторін границь записують обов’язково після задання границі всім сторонам блоку.

Допустимо задавати індивідуальні стилі, колір та ширину для різних сторін елемента.

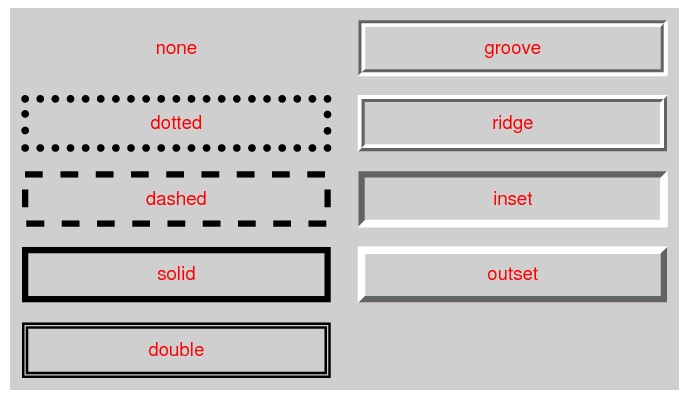
Якщо є лише одне значення компонента, воно застосовується до всіх сторін. Якщо є два значення, для верхнього та нижнього встановлюється перше значення, а для правого та лівого – друге. Якщо є три значення, для верхнього встановлюється перше значення, для лівого і правого – друге, а для нижнього – третє. Якщо є чотири значення, вони застосовуються до верхнього, правого, нижнього та лівого відповідно.

# border-color

Властивість border-color – це скорочена властивість для встановлення кольору границі border-top-color, border-right-color, border-bottom-color і border-left-color в одній декларації.

# border-style

Властивість border-style – це скорочена властивість для встановлення стилю границі border-top-style, border-right-style, border-bottom-style і border-left-style в одній декларації. Значення border-style:

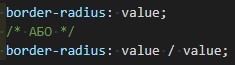


# border-width

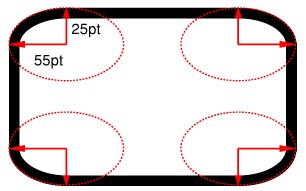
Властивість border-width – це скорочена властивість для встановлення ширини границі border-top-width, border-right-width, border-bottom-width і border-left-width в одній декларації.

# border-radius

Встановлює радіус заокруглення кутів блоку.

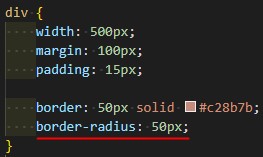


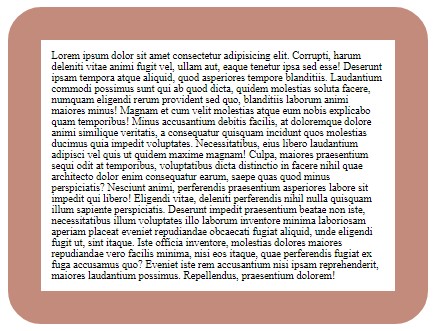
Два значення value властивості border-radius визначають радіуси чверті еліпса, який визначає форму кута зовнішнього краю границі (див. рис. нижче). Перше значення – горизонтальний радіус, друге – вертикальний радіус. Якщо друге значення опущено, воно копіюється з першого. Якщо будь-яка довжина дорівнює нулю, кут квадратний, а не закруглений. Відсотки для горизонтального радіуса відносяться до ширини границі, тоді як відсотки для вертикального радіуса стосуються висоти границі. Від’ємні значення недійсні.

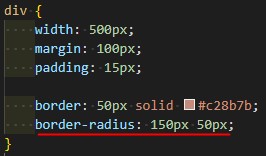


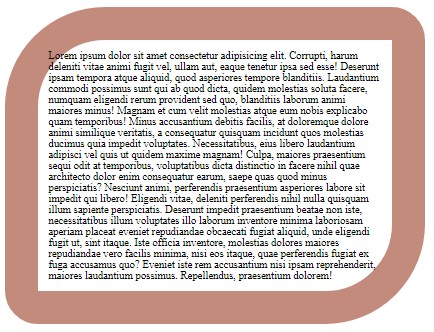
Скорочення border-radius встановлює для всіх чотирьох сторін границі заокруглення кутів. Якщо значення вказано перед і після «слешу», тоді значення перед – встановлюють горизонтальний радіус, а значення після – вертикальний радіус. Якщо «слеш» відсутній, то значення встановлюють обидва радіуси однаково. Чотири значення для кожного радіуса подано в такому порядку: верхній лівий, верхній правий, нижній правий, нижній лівий. Якщо нижній лівий кут опущено, це те саме, що верхній правий. Якщо нижній правий кут опущено, це те саме, що верхній лівий. Якщо верхній правий кут опущено, це те саме, що верхній лівий.

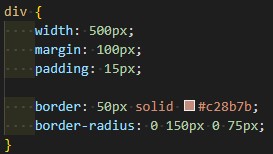
Приклади:

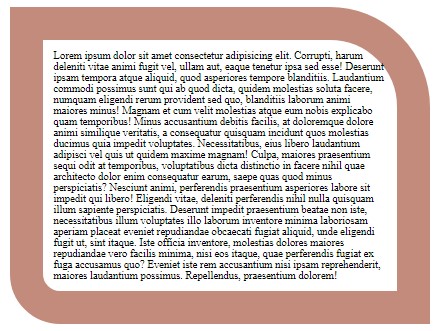


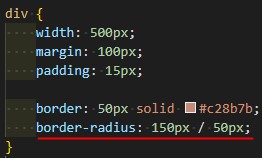


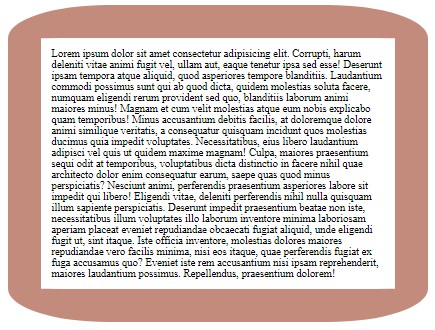


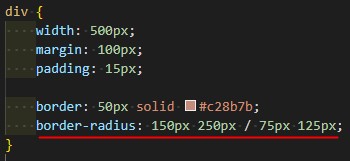


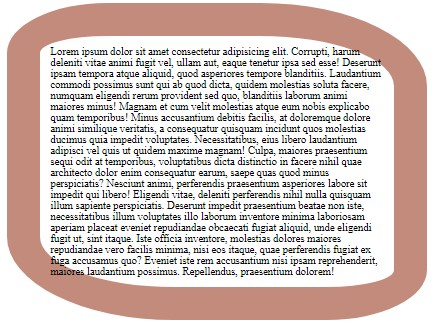












## *Формування кутів*

Радіус краю заповнення (внутрішньої границі) – це радіус зовнішньої границі мінус відповідна товщина границі. У випадку, коли це призводить до від’ємного значення, внутрішній радіус дорівнює нулю (у таких випадках його центр може не збігатися з центром кривої зовнішньої границі). Подібним чином радіус краю вмістимого дорівнює радіусу краю заповнення мінус відповідне заповнення або, якщо воно від’ємне, дорівнює нулю. Таким чином, товщина границь і відступів у вигнутій області інтерполюється з суміжних сторін, і коли дві суміжні границі мають різну товщину, у куті відображатиметься плавний перехід між товщою та тоншою границями.

Всі стилі границь (solid, dotted, inset, etc.) повторюють криву границі.

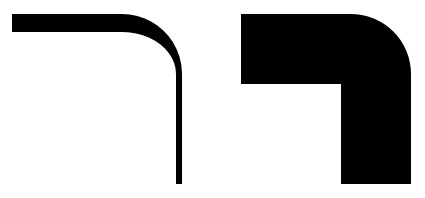
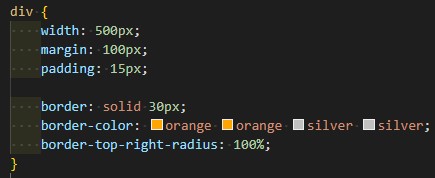
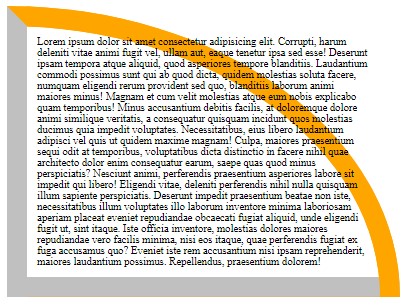


Рисунок – Ефект закругленого кута, коли дві границі, які він з’єднує, мають різну товщину

(ліворуч), і ефект заокругленого кута на межах, товщі за радіус кута (праворуч)

Приклад. Там, де крива границі-радіуса поширюється на границі протилежних сторін, дуга краю заповнення менше 90°.



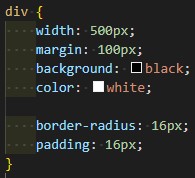


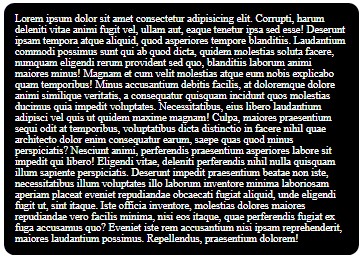
## *Обрізання кутів*

Хоча зображення границь не впливають на border-radius, інші ефекти (прорисовка, padding, контентний відступ) повинні «обрізатись» до відповідних кутів.

Приклад: додає відповідні відступи, щоб вмістиме блоку не виходило за кути.

Зауважте, що границі немає, але фон все одно матиме закруглені кути.

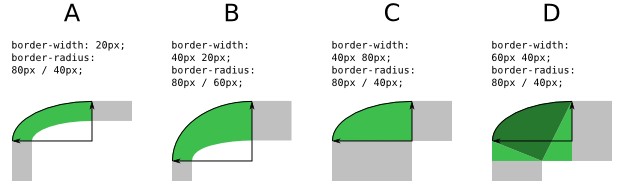




## *Перехід кольорів і стилів*

Переходи кольорів і стилів мають міститися в межах сегмента границі, який перетинає найменший прямокутник, що містить обидва радіуси границі, а також центр внутрішньої кривої (який може бути точкою, що представляє кут краю заповнення, якщо радіуси границі менші за ширину границі).

Якщо одна з цих границь має нульову ширину, то інша займає всю перехідну область. В іншому випадку центр переходів кольору та стилю між суміжними границями є точкою вздовж кривої, яка є безперервною монотонною функцією співвідношення ширини границь. Однак не визначено, як виглядають ці переходи або яка функція відображає це співвідношення в точку на кривій.



Враховуючи ці форми кутів, переходи кольорів і стилів повинні міститися в зеленій області. У випадку D прямокутник, визначений радіусами границі, не включає центр внутрішньої кривої (який є гострим кутом), тому область переходу розширюється, щоб включити цей кут. Переходи можуть займати всю область переходу, але не обов’язково, наприклад, градієнтний колірний перехід між двома стилями суцільної границі може займати лише область, обмежену кінчиками зовнішніх радіусів і кінчиками внутрішніх радіусів (представлено у випадку D темно-зеленою областю).

Властивість border-radius може працювати без поєднання із властивістю border. Проте, щоби побачити результат, нам необхідно спочатку оголосити властивість overflow: hidden; (ця властивість скриває все, що знаходиться за межами границі), і тоді результат буде видимий.



Дуже часто border-radius та overflow використовують для формування фігури круг:



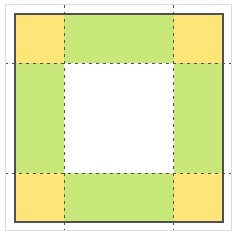
Властивість border-radius не застосовна до рядкових тегів, таких як span, проте застосовна до тегу img, навіть без спільного поєднання із властивістю overflow: hidden;

# border-image

Властивість, яка дозволяє встановити фонове зображення для границі елемента. Властивість заповнює границю заданою картинкою, розподіляючи частини зображення таким чином, щоб кутові частини знаходилися в кутах границі, а простори між ними заповнювалися рештою частин зображення.

При цьому border-radius не впливає на border-image, тому круглі картинки для границі – єдиний спосіб зробити округлу границю, головна вимога – симетричність зображень.

Границя із зображенням повністю «гумова», і може бути додана об'єкту будь-якого розміру. Для border-image обов'язково необхідно задавати зображення (border-image-source), розмір кутової частини (border-image-slice) та повторення картинки (border-image-repeat). Задана картинка розбивається за такою схемою:



Жовтим виділено кутові частини, розмір яких визначається властивістю border-imageslice, зеленим – заповнюючи частини.

*border-image-source* Можливі значення: none або <image>. Розмір картинки дорівнює товщині рамки. Колір та стиль рамки ігноруються. Якщо задати лише border-image-source, картинка заповнить собою лише кути.

*border-image-slice*

Важлива властивість, яка визначає розмір шматка зображення, що заповнить кути границі. Інші частини будуть використані для заповнення простору між кутами за алгоритмом, заданим у border-image-repeat.

Можливі значення:

* <відсотки> – розраховуються щодо розміру зображення. Горизонтальні щодо ширини, вертикальні щодо висоти.
* <числа> – пікселі (для растрового зображення) або координати (для векторного). Одиниці виміру не вказуються.
* fill – ключове слово, яке доповнює попередні значення. Якщо воно задано, зображення не обрізається внутрішнім краєм границі, а заповнює область всередині границі. Дуже корисно для округлих рамок.

Щоб визначити значення для кожної сторони, можна встановити кілька значень через пробіл. Сума значень протилежних сторін повинна бути меншою за розмір картинки, інакше не буде чим заповнювати простір між кутами.

*border-image-repeat*

Властивість визначає, як будуть заповнюватися проміжки між кутами. Можливі значення:

* stretch – розтягує заповнюючу ділянку картинки. Значення за замовчуванням;
* repeat – повторює заповнюючу ділянку, при цьому видно місця стиків з кутовим малюнком;
* round – заповнює проміжок між кутами. Може бути помітний стик у середині сторони.

Можна задати два значення відразу, перше відповідатиме за поведінку картинки у верхній та нижній частини границі, друге – за ліву та праву частини границі.

*border-image-width*

Властивість керує шириною видимої частини рамки, масштабує її. Якщо це значення більше за ширину border-width, картинка рамки заповзе під вміст, навіть якщо не задана властивість fill.

Можливі значення:

* <довжина> – значення в px або em;
* <%> – значення у відсотках щодо розміру зображення;
* <числа> – числове значення, на яке множиться border-width
* auto – ключове слово. Якщо воно задано, значення дорівнює відповідному borderimage-slice. Якщо відповідного розміру немає, використовується значення borderwidth, при цьому зображення заповнює весь кут рамки, заповзаючи під контент.

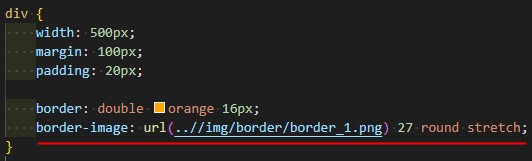
*border-image-outset*

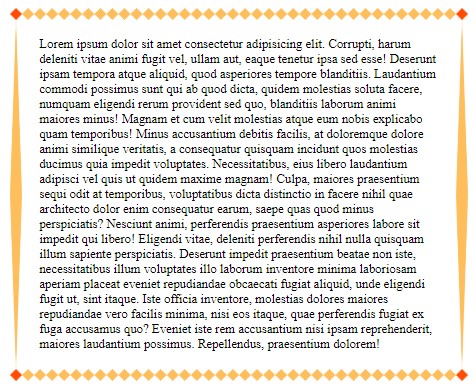
Властивість, що дозволяє відсунути рамку за межі елемента. Негативні значення не підтримуються.

Можливі значення:

* <довжина> – значення в px або em;  <числа> – числове значення, на яке множиться border-width.

Приклад. У цьому прикладі створено верхню та нижню границі, що складаються з цілої кількості помаранчевих ромбів, ліву та праву границі – з одного розтягнутого ромба. По кутах – ромби іншого кольору.

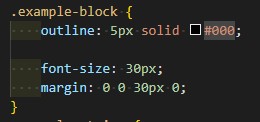




# outline

Властивість, що одночасно встановлює колір, стиль і товщину зовнішньої границі для всіх чотирьох сторін блоку. Синтаксис такий же як і в border.

На відміну від border – яка встановлюється всередині блоку та зсуває його вмістиме всередину, outline – встановлюється ззовні блоку (за межами блоку) та не впливає на вмістиме й положення блоку та його ширину.



Не можна встановити параметри outline на окремих сторонах елемента, оскільки outline застосовується одразу до всіх чотирьох сторін.

Із outline не рекомендують застосовувати властивість border-radius.

Властивість outline застосовна до рядкових тегів, проте «обгортання» границею буде залежати від вмістимого цих тегів.

## 2. Форматування box-shadow

Додає тінь до блоку. Ми вже розглядали подібну властивість text-shadow для тексту. Тут майже ідентичне, тільки для блоку. І так само як і властивість outline, box-shadow встановлюється ззовні блоку (за межами блоку). Синтаксис:



Тут:

1. ключове слово inset встановлює тінь усередині елемента;
2. зсув тіні по горизонталі (5px – вправо, -5px – вліво);
3. зсув по вертикалі (5px – вниз, -5px – вгору);
4. радіус розмиття тіні (0 – різка тінь); 5. розтягнення тіні (5px – розтяг, -5px – стиск);

6. колір тіні.

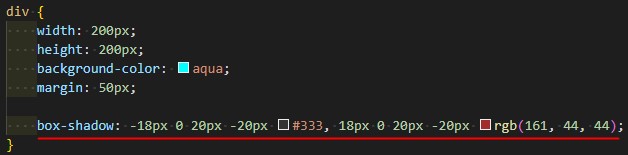
Обов'язково слід вказати лише зсув по горизонталі та вертикалі, всі інші параметри прийматимуться за замовчуванням. В цьому випадку тінь буде різкою без розмиття та чорного кольору.

За рахунок комбінацій різних параметрів та їх значень можна набути широкого різноманіття видів тіней.

* box-shadow: 5px 5px; – Різка тінь праворуч та знизу.
* box-shadow: -5px -5px; – Різка тінь зліва та зверху.
* box-shadow: 0 0 5px; – Розмита тінь навколо елемента.
* box-shadow: 0 0 5px 3px; – Розширення тіні на 3px.
* box-shadow: 0 0 5px 2px red; – Червоне світіння навколо елемента.
* box-shadow: 0.4em 0.4em 5px rgba(122,122,122,0.5); – Напівпрозора тінь.
* box-shadow: inset 0 0 6px; – Тінь усередині.

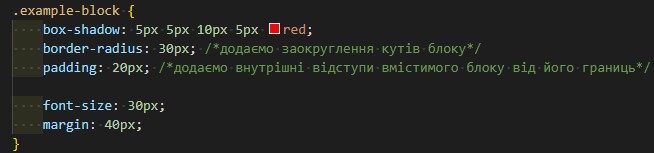
Як видно, зсув тіні не обов'язково задавати в пікселах, хоча це зручно. Колір тіні можна вказувати в будь-якому доступному форматі, так для отримання напівпрозорої тіні підійде формат RGBA.

Тінь у елемента може бути не одна, а відразу декілька, їх параметри перераховуються через кому.



Перша тінь відображається зліва від блоку з радіусом розмиття 20px, її розмір зменшено за рахунок четвертого параметра (-20px). Параметри другої тіні вказуються після коми, тінь відображається праворуч від блоку і також зменшена для симетрії.

Із box-shadow можна застосовувати властивість border-radius.



Властивість box-shadow застосовна до рядкових тегів, проте це не завжди той результат на який ми очікували.

## 3. «Плаваючі» елементи позиціонування float

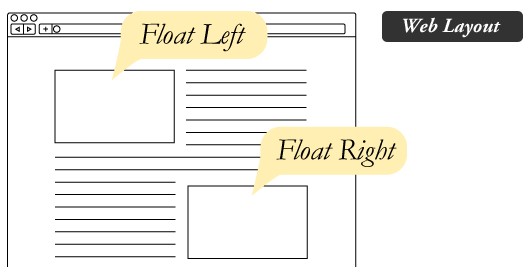
Деякі елементи CSS є блоковими, тому починаються з нового рядка, інші елементи є «рядковими», тому відображаються на сторінці в один рядок.

Один із способів перепризначити елементам тип «обтікання» – це використання властивості float – властивість CSS для позиціонування елементів, яка визначає по якій стороні буде вирівнюватись елемент, при цьому інші елементи будуть «обтікати» його з іншої сторони.

Усього є 4 значення для властивості float.

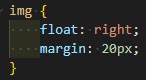
* left – вирівнює елемент по лівому краю, всі інші елементи «обтікають» його справа.
* right – вирівнює елемент по правому краю, всі інші елементи «обтікають» його зліва.
* none (за замовчуванням) – забезпечує, що елемент не «плаватиме».
* inherit – поведінка має бути такою ж, як і у батьківського елемента.

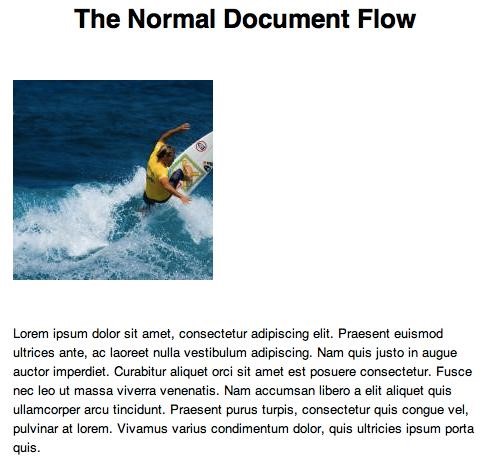
Класичний приклад – використання float для вирівнювання картинки з лівого або правого краю сторінки, яку «обтікає» текст.



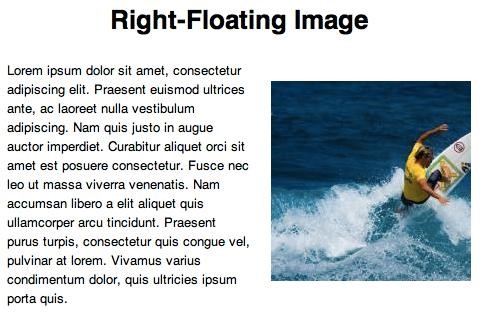
Простий HTML-код картинки та абзацу, які відображаються з нового рядка:







Виходить вирівнювання з правого краю:



Якщо тексту більше, то абзац буде «обтікати» картинку:



Припустимо, нам потрібно зробити відступ у 20px між картинкою та текстом. Ось така конструкція не працюватиме:



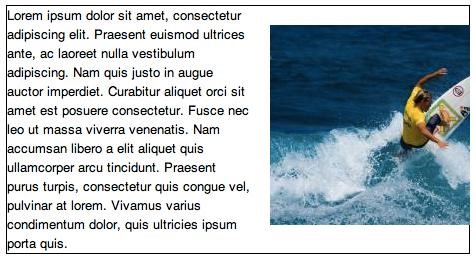
Правильно ось так:



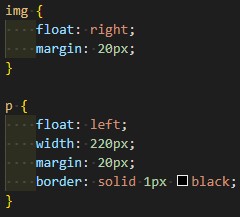


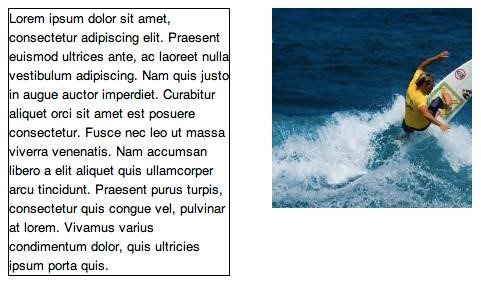
Чому ж не працює відступ для абзацу? Щоб розібратися, додамо рамку:





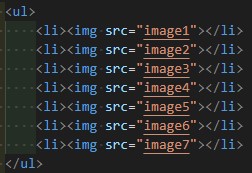
Тому, що картинка знаходиться всередині абзацу. Тому і не працює властивість margin у першому випадку. Щоб виправити це, можна застосувати float:left до абзацу та вказуємо абсолютну ширину:



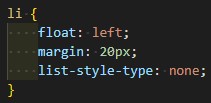


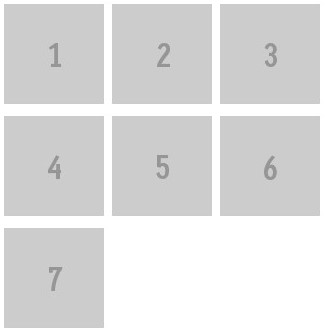
Перейдемо до складніших варіантів використання float: до правил, що регулюють «плаваючі» об'єкти. Це буває необхідним при верстці галереї зображень.

Наприклад:

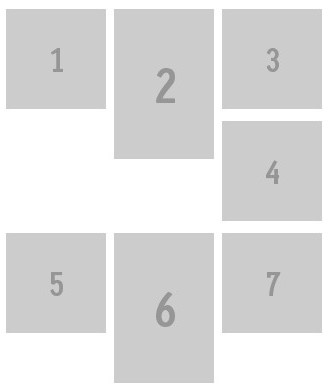


За замовчуванням кожен елемент списку з'явиться з нового рядка. Якщо застосувати до кожного float:left, зображення стануть в один ряд із перенесенням рядка:

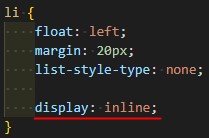


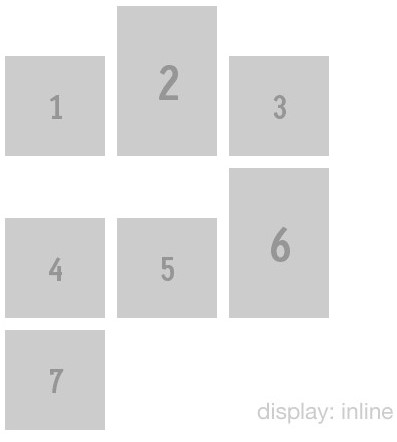


Але якщо зображення різної висоти?

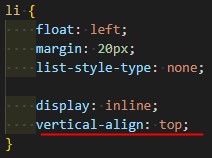


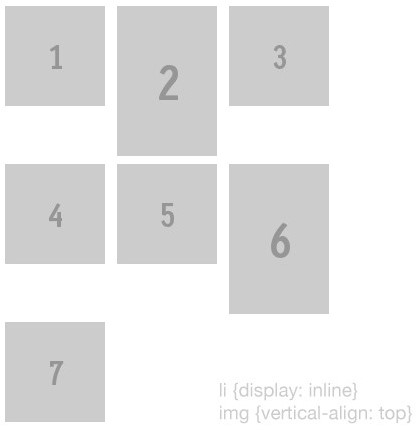
Якщо ми додамо елементам списку відображення в один рядок, вийде:



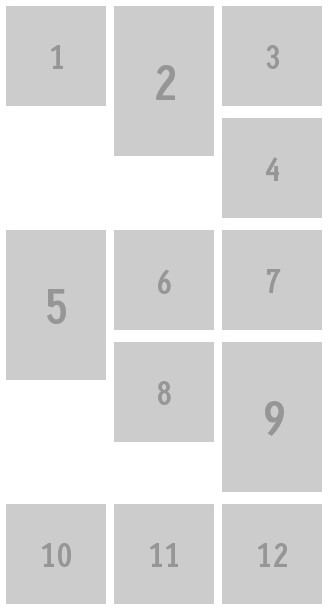


А тепер вирівнюємо по вертикалі:



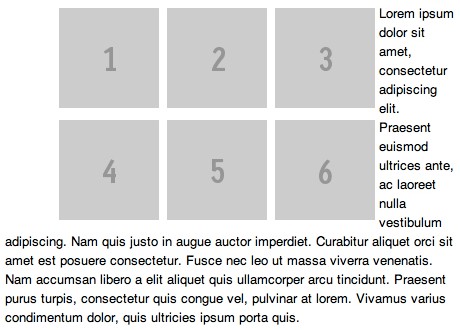


Слід пам'ятати, що у разі більшої висоти картинок, інші зображення обтікають лише одне попереднє, наприклад:



За допомогою float зручно групувати елементи на сторінці, але великою проблемою стає те, що наступні елементи (текст або блок) також одержують властивість обтікання.

Наприклад, у нас є блок картинок, текст під яким починає «обтікати» весь блок:



# clear

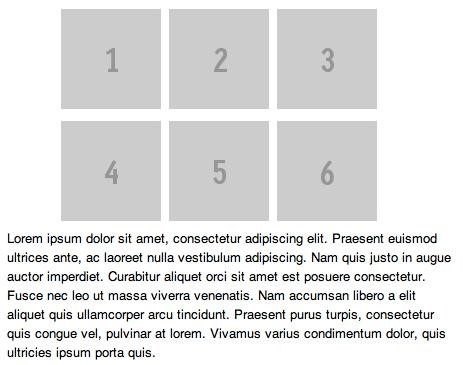
Щоб уникнути попереднього результату, необхідно використовувати властивість clear – встановлює з якого боку елемента заборонено його «обтікання» іншими елементами.

Властивість clear може набувати чотирьох значень.

* none – скасовує дію самої властивості clear, в результаті зміст тега (наприклад, текст) обтікає вебелемент відповідно до заданих стилів у вигляді float;
* left і right – ліквідує обтікання відповідно до лівого чи правого краю;
* both – перешкоджає обтіканню одночасно праворуч та ліворуч. Це найпоширеніше значення clear у практичному використанні.

Приклад: Потрібно застосувати clear:both до абзацу:



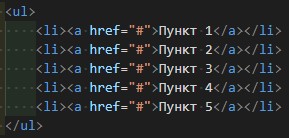


Дев’ять правил float-елементів:

* Плаваючі елементи не можуть виходити за край свого контейнера-батька.
* Кожен плаваючий елемент буде відображатися праворуч або нижче попереднього при float:left, або зліва і нижче, при float:right.
* Блок із float:left не може бути правішим, ніж блок із float:right.
* Плаваючий елемент не може виходити за межі верхньої межі контейнера.
* Плаваючий елемент не може розташовуватися вище, ніж батьківський блок або попередній плаваючий елемент.
* Плаваючий елемент не може розташовуватися вище, ніж попередній рядок inlineелементів.
* Плаваючий блок повинен бути розташований якомога вище.
* Один плаваючий елемент, що йде за іншим, не може виходити за межі свого контейнера – відбувається перенесення на наступний рядок.
* Блок з float:left має бути розташований якомога лівіше, а з float:right – якомога правіше.

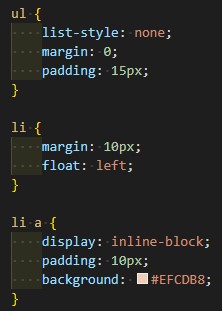
## 4. Горизонтальне меню за допомогою CSS (float)

Далі розберемо цілком конкретне завдання щодо створення горизонтального меню за допомогою плаваючих блоків, яке часто використовується при розробці вебсайтів.



CSS-код

Приберемо маркери навпроти кожного з пункту, за допомогою list-style: none, перетворимо рядкові теги гіперпосилань на блокові, застосувавши до них правило display: inline-block, додамо кожній з них фон, задамо зовнішні відступи margin, щоб відсунути пункти меню один від одного, далі необхідно пункти меню розташувати вертикально через float: left.





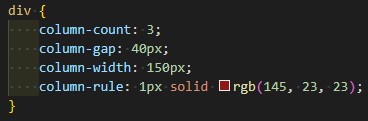
**5. Багатоколоночний текст** CSS3 пропонує відразу чотири властивості для створення та управління колонками:

column-count, column-gap, column-width та column-rule. Для наочності значення деяких властивостей показано на рис.1.



* column-count – встановлює оптимальне число колонок.
* column-gap – відстань між колонками.
* column-width – оптимальна ширина колонок.
* column-rule – розділова лінія між колонками.

Слід пояснити, що таке «оптимальне число» та «оптимальна ширина». Властивість column-count задає не точну кількість колонок, а бажане. Браузер намагатиметься розбити текст саме на стільки колонок, але при зменшенні розмірів вікна скорочуватиме кількість колонок, щоб зберегти комфортність читання. Аналогічно і з шириною. Властивість columnwidth вказує на мінімальну ширину колонки, при цьому браузер це значення може проігнорувати, якщо розмір вікна недостатній для забезпечення заданої ширини колонок.



# ПРАКТИЧНА ЧАСТИНА

Початкові налаштування:

* всі елементи кожної html-сторінки поміщені у блочний елемент «обгортка (оболонка)»;
* ширина блоку «обгортка» 1140px;
* вирівняти всі елементи в блоці «обгортка» по центру відносно вікна браузера;
* внутрішні поля – 20px;

Інші налаштування:

* всі рисунки на вибір студента;
* стилі тексту та фону, розміри рамок та тіней повинні відповідати 85% наведеним прикладам;
* розміщення зображень, блоків та пунктів меню реалізовувати через float & clear позиціювання;

Структура кожної html-сторінки:  блочний елемент header;

* блочний елемент main;

Структура елементу header:

* блочний елемент div (логотип), який містить рисунок-посилання на index.html (завдання 1);
* блочний елемент div (горизонтальне меню), який містить маркерований список із п’яти пунктів-посилання на html-сторінки кожного завдання із відносними адресами; **Завдання 1 – Багатоколоночний текст + float позиціювання картинок**
* оптимальне число колонок – 3;
* відстань між колонками – 35px;
* розділова лінія між колонками: товщина лінії – 5px, стиль – пунктирна; колір – червоний;
* ширина кожного рисунку – 200px;
* зовнішні відступи від рисунка до «обтікаючого» тексту: коливаються від 10px до 20px. Зверстати сторінку за прикладом – сторінка index.html (Завдання 1):

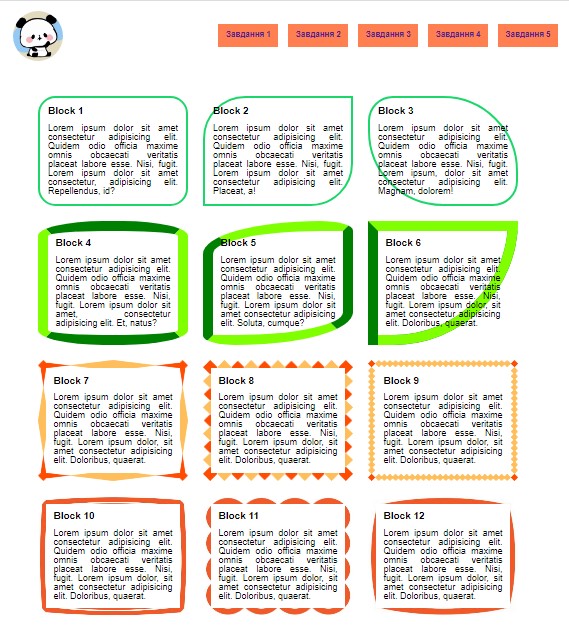


**2 – Заокруглені рамки та зображення для рамок через властивість border +**

# float позиціювання блоків

* створіть 12 змістових блоків за зразком;
* ширина блоку – 300px;
* задайте властивості заокруглених кутів (border-radius) для перших шести та властивості зображень для границь для останніх шести блоків.

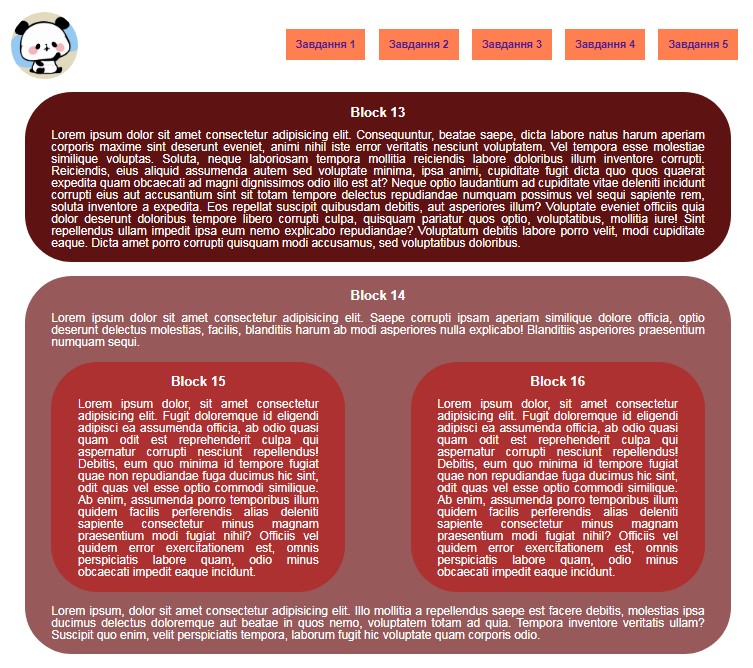
Зверстати сторінку за прикладом – сторінка page\_2.html (Завдання 2):



**3 – Поєднання властивостей border-radius та background без застосування**

# border + float позиціювання блоків

Зверстати сторінку за прикладом – сторінка page\_3.html (Завдання 3):

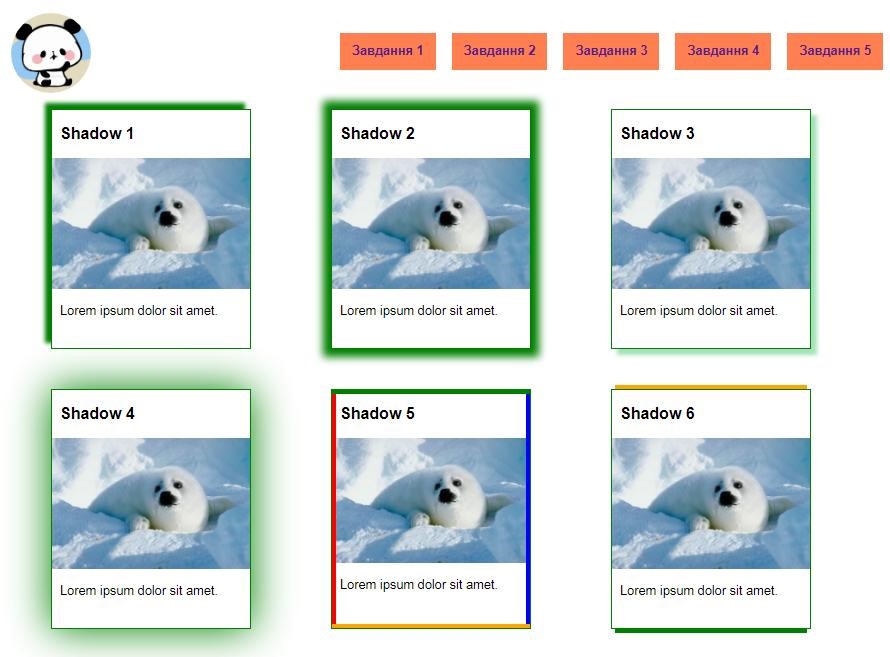


**4 – Поєднання властивостей border-radius та background без застосування**

# border + float позиціювання блоків

* створіть 6 змістових блоків за зразком;
* ширина блоку – 250px;
* висота блоку – 300px;
* задайте границю: товщина лінії – 1px, стиль – суцільна, колір – зелений;
* задайте властивості box-shadow для кожного блоку та отримайте результат наведений на рисунку.

Зверстати сторінку за прикладом – cторінка page\_4.html (Завдання 4):



**5 – Поєднання значень лінійного та радіального градієнтів + float**

# позиціювання блоків

* створіть 5 змістових блоків за зразком;
* ширина блоку – 300px;
* висота блоку – 400px;
* ширина та висота картинки – 200px;
* задайте для кожного блоку лінійні, радіальні та множинні градієнти та отримайте результат наведений на рисунку.

Зверстати сторінку за прикладом – сторінка page\_5.html (Завдання 5):

