Що таке WEB ?

WEB — система програмування, створена Дональдом Кнутом як перша реалізація його концепції «грамотного програмування»: ідеї, що програмне забезпечення можна створювати як літературний твір. Такий твір складається з фрагментів початкового коду, вбудований у текст, що описує програму природною мовою людини. Це пряма протилежність більшості мов програмування, де текст для читання людиною (документація) вбудовується у тіло програми.
WEB складається з двох програм: TANGLE, що генерує текст програми на Паскалі, і WEAVE, що продукує відформатовану TeX-документацію

Існує версія системи WEB для мови C, CWEB (наприклад, програма Web2C виконує ту ж роль, що і TANGLE для генерації Паскаль-тексту). З інших програм можна відзначити noweb, що не залежить від кінцевої мови програмування.

Найбільш відомі і важливі програми, написані на WEB — це TeX і Metafont.

І все? А з чого складається?

Вебсторінка Інформаційний ресурс, доступний в мережі World Wide Web, який можна переглянути у веб браузері.

Браузер Це програма, що дозволяє відображати сторінки мережі Інтернет на екрані комп'ютера, мобільного телефона, планшета, розумного годинника, передавати і отримувати дані з «всесвітньої павутини».

WEB-розробка Це процес створення WEB-сайтів або WEB-додатків.

Основними етапами процесу є:

- вебдизайн,
- верстка сторінок,
- програмування для веб на стороні клієнта і сервера,
- конфігурування вебсервера.

Front end та back end

Front end — це інтерфейс для взаємодії між користувачем і back end. Front end та back end можуть бути розподілені між однією або кількома системами.
В програмній архітектурі може бути багато рівнів між апаратним забезпеченням та кінцевим користувачем. Кожен з цих рівнів може мати як front end, так і back end. Front — це абстракція, спрощення базового компоненту через надання користувачу зручного (user-friendly) інтерфейсу.

Front - End

Це зовнішній вигляд сайту, його структура

Back - End

Це вся механіка сайту, його «невидимий двигун»

Front - End (Фронтенд)

Все, що браузер може читати, виводити на екран і / або запускати. Щоб відобразити контент сайту найкращим чином та забезпечити оптимальну взаємодію користувача з сайтом, фронтенд-розробники використовують багато корисних інструментів, найважливіші з яких це
HTML, CSS та JavaScript.

HyperText Markup Language — мова розмітки гіпертексту це код, який використовується для структурування і відображення веб-сторінки та її контенту.
HTML це не мова програмування, це мова розмітки, яка каже вашому браузеру, як відображати вміст веб-сторінки, яку ви переглядаєте.

HTML - документ має свою структуру

```
<!DOCTYPEhtml>
<html>
<head>
<title>WEB programming</title>
    <script></script>
</head>
<body>
<!–Початок коду-->
    <div>
        <!–Контент 1-->
    </div>
    <div>
        <!–Контент 2-->
    </div>
<!–Кінець коду-->
</body>
</html>
```

Тег - елемент розмітки гіпертексту. Мітка, яку використовує розробник для вказівки браузеру, як він повинен показувати WEB-сайт. Тег також має свою структуру

Всі HTML-елементи діляться на п'ять типів:

- **порожні елементи** - <area>, <base>,
, <col>, <embed>, <hr>, <img>, <input>, <link>, <menuitem>, <meta>, <param>, <source>, <track>, <wbr>;
- **елементи з неформатований текстом** - <script>, <style>;
- **елементи, які виведуть звичайний текст** - <textarea>, <title>;
- **елементи з іншого простору імен** - MathML і SVG;
- **звичайні елементи** - всі інші елементи.

Як написати і запустити HTML на комп'ютері?

Крок 1. Качаємо текстовий редактор

Для того, щоб написати код, згодиться взагалі будь-який текстовий редактор. Підійде навіть «Блокнот» на вашому комп'ютері (але в ньому дуже незручно все робити). Ми завантажили та встановили хороший редактор, заточений під веб-розробку. Покажемо все на прикладі Visual Studio Code.

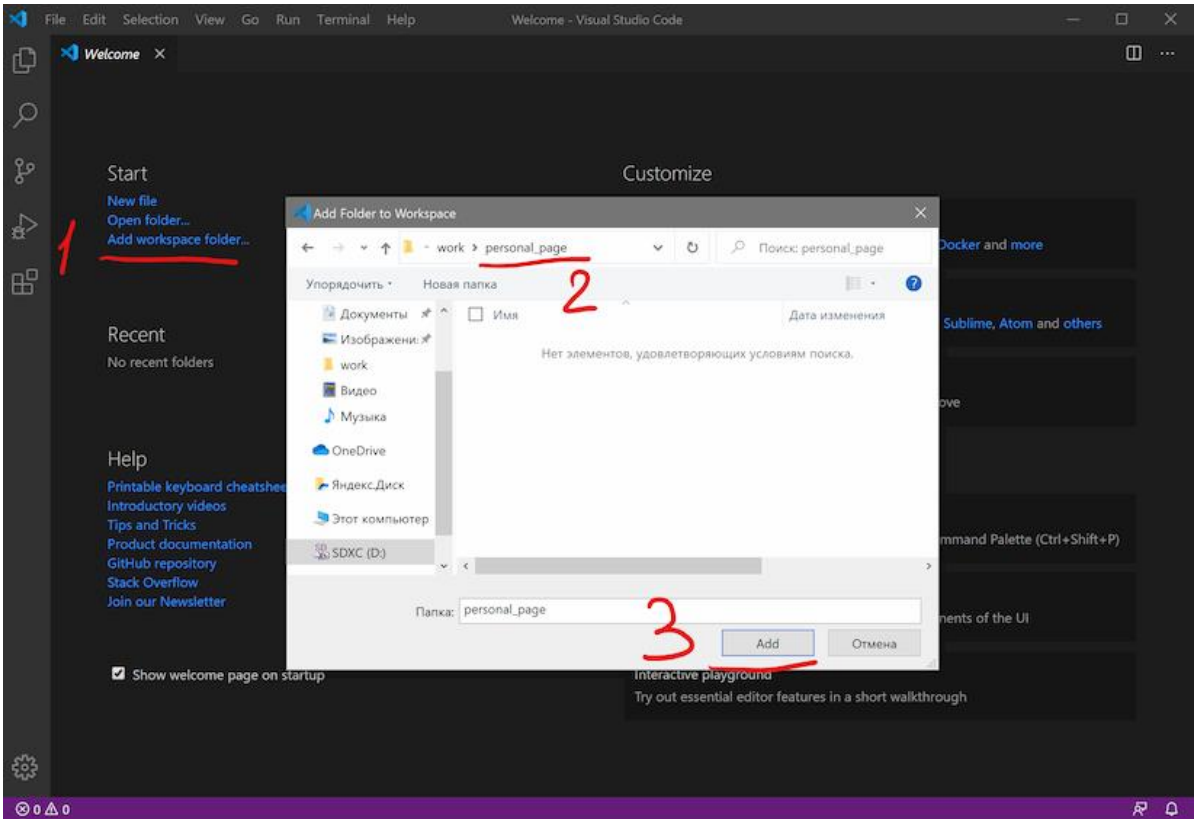
Зайдіть на https://code.visualstudio.com/ і скачайте редактор. Якщо у вас Windows, то натисніть на будь-яку з синіх кнопок. Якщо OS X або Linux - натисніть Other platforms.

Установка пройде як завжди - потрібно запустити файл VSCodeUserSetup, багато раз натиснути «Далі» і поставити пару галочок.

Крок 2. Запускаємо редактор і оглядаємося

Свіжовстановленому VS Code зустрічає нас екраном з великою кількістю посилань. З ними можна познайомитися пізніше, а зараз потрібно налаштувати все для роботи.

Добре б, щоб під час роботи всі потрібні файли лежали в одній папці (поки проект маленький, так можна робити). Для цього додамо робочу папку, щоб VS Code показував нам тільки її вміст.



По кроках на скріншоті:

1. Add workspace folder - відкриває меню вибору папки.
2. Створимо нову папку personal_page в будь-якому зручному місці і зайдемо в неї.
3. Натиснемо Add.

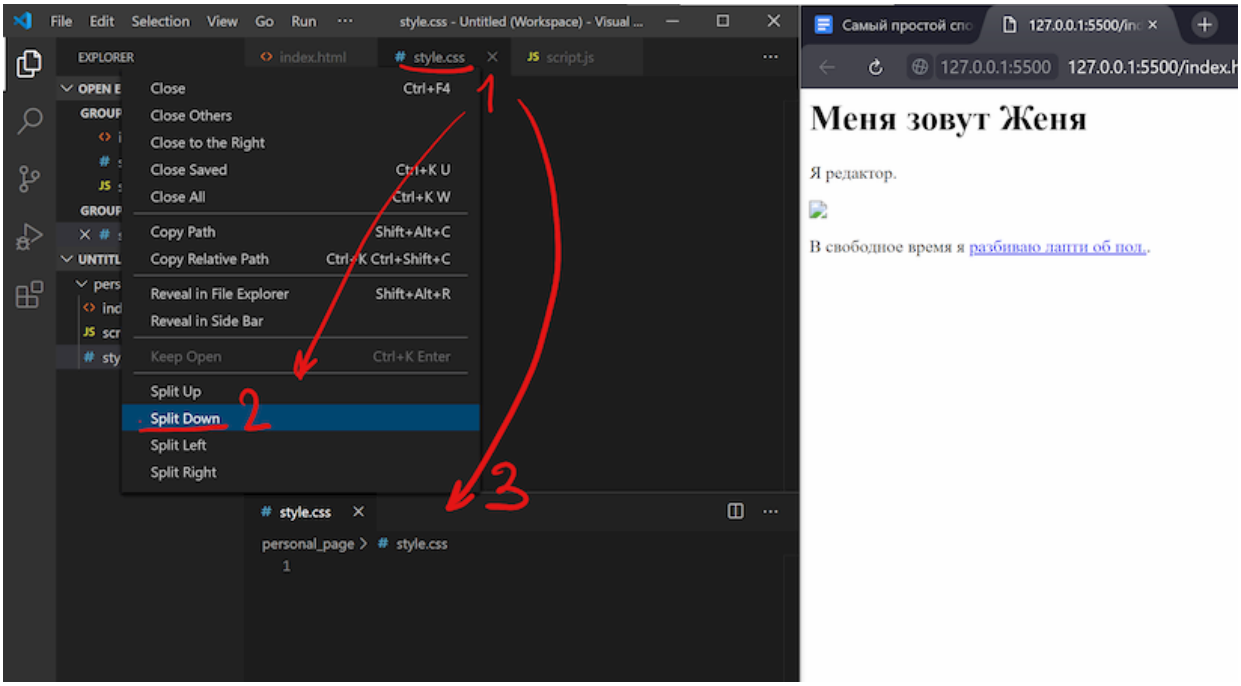
   Після цього зліва з'явиться панель Explorer з порожнім робочим простором Untitled (Workspace). Ми створили папку, давайте її наповнимо.

Крок 3. Додаємо файли

Після створення папка порожня. Клацнемо правою кнопкою по заголовку personal_page і додамо три файли, які знадобляться в роботі - index.html, style.css і script.js. Для початку цього вистачить.

Крок 4. Робимо роботу зручніше

Зараз все три файли відкриті у вкладках, і між ними не завжди зручно перемикатися. Щоб було зручніше, код зі стилями можна перенести в іншу частину вікна, наприклад, вниз. Для цього натисніть правою кнопкою по вкладці з style.css і виберіть split down, щоб побачити результат.



Крок 5. Додаємо код

Поки відредагуємо тільки index.html (файл з розміткою) і style.css (файл зі стилями), а script.js залишимо на майбутнє. Якщо у вас вже є який-небудь код, напишіть його, або використовуйте готовий - ми, наприклад, візьмемо код з інтерактивних курсів.

index.html

```
<!DOCTYPE html>
<html lang="ru">
 <head>
  <title>Сайт начинающего верстальщика</title>
  <link rel="stylesheet" href="style.css">
 </head>
 <body>
  <header>
   <nav>
    На главную
   </nav>
  </header>
  <main>
   <article>
    День первый. Как я забыл покормить кота
```

Кто бы мог подумать, что семантика это так важно, мне срочно нужно было об этом поговорить.

Взгляд упал на кота. Кот издал настойчивое «Мяу». И я понял — пришло время для первой записи в блог. И покормить кота.

```
   </article>
   <aside>
    Здесь могла быть ваша реклама.
   </aside>
```

```
    </main>
    <footer>
      Подвал сайта
    </footer>
  </body>
</html>
```

style.css

Скопіюємо код зі стилями з файлу https://htmlacademy.ru/assets/courses/299/outlines.css - відкрийте його в браузері, скопіюйте всі рядки і вставте в файл style.css в редакторі.

Крок 6. Запускаємо код і дивимося на результат

Найпростіший спосіб - відкрити папку з файлами через провідник і запустити файл index.html. Ви побачите результат верстки в браузері, але це не дуже зручно - при будь-яких змінах доведеться переходити в браузер і оновлювати сторінку.

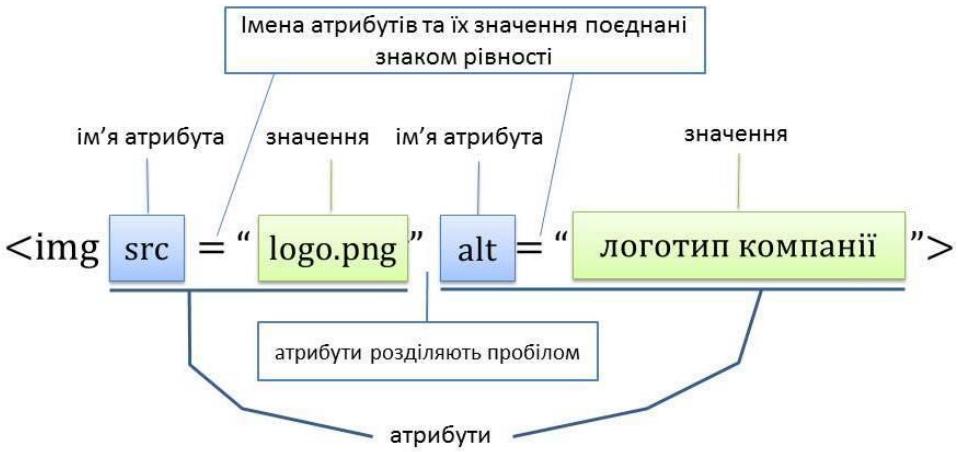
Базові теги HTML

У загальному випадку тег записується наступним чином:

```
<ім'я тега атрибут1="значення1" атрибут2="значення2" ...>
```

Ось декілька прикладів тегів з атрибутами:

```
<p class="important"> ... </p>
<a class="external" href="http://cpto.dp.ua"> ... </a>
<img class = "logo" src = "logo.png">
```



Оформлення смислових частин сторінки абзацами:

- … – розбиває текст на параграфи (абзаци);
-
 – обриває рядок і починає новий;
- <nobr>…</nobr> – забороняє перенос тексту на інший рядок;
- <wbr>…</wbr> – дозволяє розривати довгі слова (наприклад, при виведенні довгих медичних термінів);
- <!- - … - -> – тег коментарів; розміщений в ньому текст не відображається на web-сторінці.

Основні теги форматування тексту та елементів дизайну

- <hn>…</hn> – керує розміром символів в заголовках ($1 \le n \le 6$); зі збільшенням n розмір шрифту зменшується.
- <big>...</big> – збільшує розмір шрифту на 10% у порівнянні з базовим.

- <small>…</small> – зменшує розмір шрифту на 10% у порівнянні з базовим.
- <pre>…</pre> – позначає фрагмент тексту, форматування якого здійснено наперед; зручно використовувати при виведенні на екран текстів програм на мовах програмування, або при виведенні тексту, який містить спеціальні символи; в деяких випадках замінює тег <nobr>…</nobr>.
- <code>…</code> – позначає фрагмент тексту усередині рядка, який вдповідає елементам коду.
- <hr> – непарний тег горизонтальної лінії; може бути засобом організації тексту і дизайну.

  Атрибути тега <hr>:

  - align=left (right, center) – вирівнювання лінії на сторінці;
  - size=n – товщина лінії в пікселях (1≤n≤175); за замовчуванням 2 пікселі;
  - color="колір" – колір лінії;
  - width=n% – ширина лінії у відсотках до ширини екрану;
  - noshade – атрибут без параметра; показує об'ємні лінії, встановлюється за замовчуванням;
  - shade – атрибут без параметра; показує контур лінії; якщо заданий колір, то атрибут не працює.
- <left> ... </left> – вирівнювання тексту по лівому краю сторінки.
- <right> ... </right> – вирівнювання тексту по правому краю сторінки.
- ... – вирівнювання тексту по середині сторінки.
  Наведені вище теги вирівнювання дещо застаріли. Для цього краще використовувати відповідні властивості CSS.

- … – виділення напівжирним шрифтом.
- … – виділення курсивом.

HTML-списки використовуються для групування пов'язаних між собою фрагментів інформації. Існує три види списків:

- **маркований список** - - кожен елемент списку <li> зазначається маркером,
- **нумерований список** - - кожен елемент списку <li> зазначається цифрою,
- **список** Кожен елемент списку має додатковий блок, розташований збоку, який не бере участі в компонуванні.**визначень** - <dl> - складається з пар термін <dt> - <dd> визначення.

Кожен список являє собою контейнер, усередині якого розташовуються елементи списку або пари термін-визначення.
Елементи списку поводяться як блокові елементи, розташовуючись один під одним і займаючи всю ширину блоку-контейнера.

1. Маркований список

Маркований список являє собою невпорядкований список *(від англ. Unordered List)* . Створюється за допомогою елемента <ul>. В якості маркера елемента списку виступає позначка, наприклад, зафарбований кружок.
Браузери за замовчуванням додають наступне форматування блоку списку:

```
ul {
  padding-left: 40px;
  margin-top: 1em;
  margin-bottom: 1em;
}
```

CSS
Кожен елемент списку створюється за допомогою елемента <li> *(від англ. List Item)* .

Для елемента доступні глобальні атрибути .

```
<ul>
  <li>Microsoft</li>
  <li>Google</li>
  <li>Apple</li>
  <li>IBM</li>
</ul>
```

HTML

- Microsoft
- Google
- Apple
- IBM

ФІГУРА. 1. МАРКОВАНИЙ СПИСОК

2. Нумерований список

Нумерований список створюється за допомогою елемента <ol>. Кожен пункт списку також створюється за допомогою елемента <li>. Браузер нумерує елементи по порядку автоматично і якщо видалити один або кілька елементів такого списку, то інші номери будуть автоматично перераховані. Блок списку також має стилі браузера за замовчуванням:

```
ol {
  padding-left: 40px;
  margin-top: 1em;
  margin-bottom: 1em;
}
```

CSS

Для елемента доступний атрибут value, який дозволяє змінити номер за умовчанням для вибраного елементу списку. Наприклад, якщо для першого пункту списку задати <li value="10">, то інша нумерація буде перерахована щодо нового значення.

Для елемента доступні наступні атрибути:

ТАБЛИЦЯ 1. АТРИБУТИ ЕЛЕМЕНТА <OL>

Атрибут	**Опис, прийняте значення**
reversed	Атрибут reversedзадає відображення списку в зворотному порядку (наприклад, 9, 8, 7 ...).
start	Атрибут startзадає початкове значення, від якого піде відлік нумерації, наприклад, конструкція <ol start="10">першого пункту присвоїть порядковий номер «10». Також можна одночасно ставити тип нумерації, наприклад, <ol type="I" start="10">.
type	Атрибут typeзадає вид маркера для використання в списку (у вигляді букв або цифр). Прийняті значення: 1- значення за замовчуванням, десяткова нумерація. A- нумерація списку в алфавітному порядку, заголовні букви (A, B, C, D). a- нумерація списку в алфавітному порядку, малі літери (a, b, c, d). I- нумерація римськими великими цифрами (I, II, III, IV). i- нумерація римськими малими цифрами (i, ii, iii, iv).

```
<ol>
  <li>Microsoft</li>
  <li>Google</li>
  <li>Apple</li>
  <li>IBM</li>
</ol>
```

HTML

1. Microsoft

2. Google
3. Apple
4. IBM

ФІГУРА. 2. НУМЕРОВАНИЙ СПИСОК

3. Список визначень

Списки визначень створюються за допомогою елемента <dl>. Для додавання терміна застосовується елемент <dt>, а для вставки визначення - елемент <dd>.

Блок списку визначень має наступні стилі браузера за замовчуванням:

```
dl {
  margin-top: 1em;
  margin-bottom: 1em;
}
```

CSS

Для елементів <dl>, <dt>і <dd>доступні глобальні атрибути .

```
<dl>
 <dt>Режиссер:</dt>
  <dd>Петр Точилин</dd>
 <dt>В ролях:</dt>
  <dd>Андрей Гайдулян</dd>
  <dd>Алексей Гаврилов</dd>
  <dd>Виталий Гогунский</dd>
  <dd>Мария Кожевникова</dd>
</dl>
```

HTML

режисер:
Петро Точилін
У ролях:
Андрій Гайдулян
Олексій Гаврилов
Віталій Гогунський
Марія Кожевнікова

ФІГУРА. 3. СПИСОК ВИЗНАЧЕНЬ

4. Як створити вкладений список

Найчастіше можливостей простих списків не вистачає, наприклад, при створенні змісту ніяк не обійтися без **вкладених пунктів** . Розмітка для вкладеного списку буде наступною:

```
<ul>
 <li>Пункт 1.</li>
 <li>Пункт 2.
  <ul>
   <li>Подпункт 2.1.</li>
    <li>Подпункт 2.2.
     <ul>
      <li>Подпункт 2.2.1.</li>
      <li>Подпункт 2.2.2.</li>
      </ul>
    </li>
   <li>Подпункт 2.3.</li>
  </ul>
 </li>
 <li>Пункт 3.</li>
</ul>
```

HTML

- Пункт 1.
- Пункт 2.

- Підпункт 2.1.
- Підпункт 2.2.
    - Підпункт 2.2.1.
    - Підпункт 2.2.2.
- Підпункт 2.3.

- Пункт 3.

ФІГУРА. 4. ВКЛАДЕНИЙ СПИСОК

5. Багаторівневий нумерований список

Багаторівневий список використовується для відображення елементів списку на різних рівнях з різними відступами. Розмітка для багаторівневого нумерованого списку буде наступною:

```
<ol>
  <li>пункт</li>
  <li>пункт
    <ol>
      <li>пункт</li>
      <li>пункт</li>
      <li>пункт
        <ol>
          <li>пункт</li>
          <li>пункт</li>
          <li>пункт</li>
        </ol>
      </li>
      <li>пункт</li>
    </ol>
  </li>
  <li>пункт</li>
  <li>пункт</li>
</ol>
```

HTML

Така розмітка за замовчуванням створить для кожного вкладеного списку нову нумерацію, яка починається з одиниці. Щоб зробити вкладену нумерацію, потрібно використовувати такі властивості:

- counter-reset скидає один або кілька лічильників, задаючи значення для скидання;
- counter-incrementзадає значення приросту лічильника, тобто з яким кроком нумеруватиметься кожний наступний пункт;
- content - генерується вміст, в даному випадку відповідає за виведення номера перед кожним пунктом списку.

```
ol {
  /* убираем стандартную нумерацию */
  list-style: none;
  /* Идентифицируем счетчик и даем ему имя li. Значение счетчика не указано - по умолчанию оно равно 0 */
  counter-reset: li;
}
li:before {
  /* Определяем элемент, который будет нумероваться — li. Псевдоэлемент before указывает, что содержимое, вставляемое при помощи свойства content, будет располагаться перед пунктами списка. Здесь же устанавливается значение приращения счетчика (по умолчанию равно 1). */
  counter-increment: li;
  /* С помощью свойства content выводится номер пункта списка. counters() означает, что генерируемый текст представляет собой значения всех счетчиков с таким именем. Точка в кавычках добавляет разделяющую точку между цифрами, а точка с пробелом добавляется перед содержимым каждого пункта списка */
  content: counters(li, ".") ". ";
}
```

CSS

1. пункт
2. пункт
    1. пункт
    2. пункт
    3. пункт
        1. пункт
        2. пункт
        3. пункт
    4. пункт
3. пункт
4. пункт

CSS

Структура WEB проекту

- **Project (Root) Folder:** основна папка, містить всі інші папки та файли
- **CSS Folder:** усі файли, що відносяться до стилів (.css файли)
- **Images Folder:** усі зображення (.jpg, .png, .gif, etc)
- **JavaScript Folder:** усі JavaScript файли (.js)

Термінал на Linux

ctrl + alt + T - launch terminal

Основні команди

clear - Очистити консоль

Навігація

pwd- Показати поточний каталог
ls - Показати файли в цій папці, крім прихованих
ls -f- Показати файли в цій папці, включаючи і приховані
cd c:/- Перейти в конкретний каталог
cd -- Повернутися назад
cd ..- Вийти на 1 рівень вгору
cd ../..- Вийти на 2 рівня вгору

створення каталогів

mkdir- Створити папку
cd !$- Перейти в тільки що створену папку
mkdir -p {app1,app2}- Створити відразу кілька папок
mkdir -p app/{css,js}- Створити відразу кілька вкладених папок

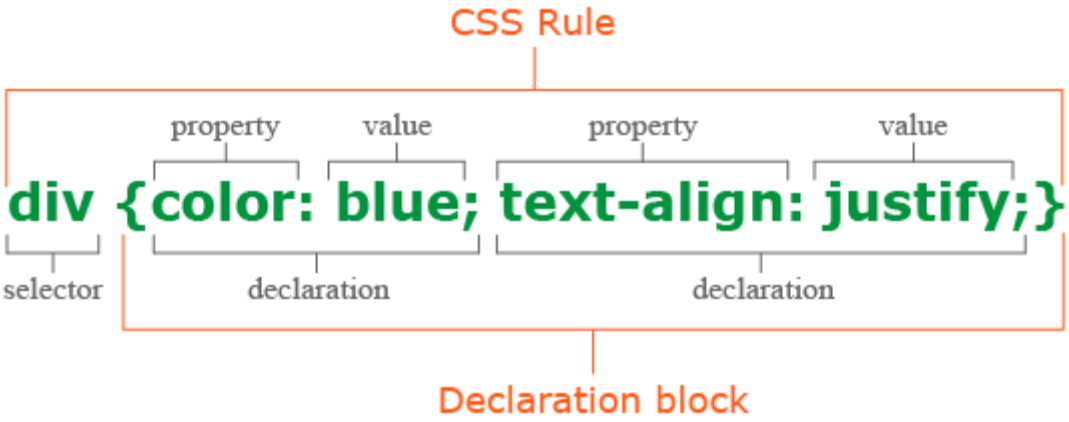
створення файлів

touch index.html- Створити файл index.html
touch app/{css/main.css,js/main.js,index.html}- Створити відразу кілька файлів, ніяких зайвих прогалин бути не повинно

видалення файлів
touch- дозволяє створювати файли
rm test- Видалити порожню папку test
rm -r test- Видалити папку test з файлами всередині неї

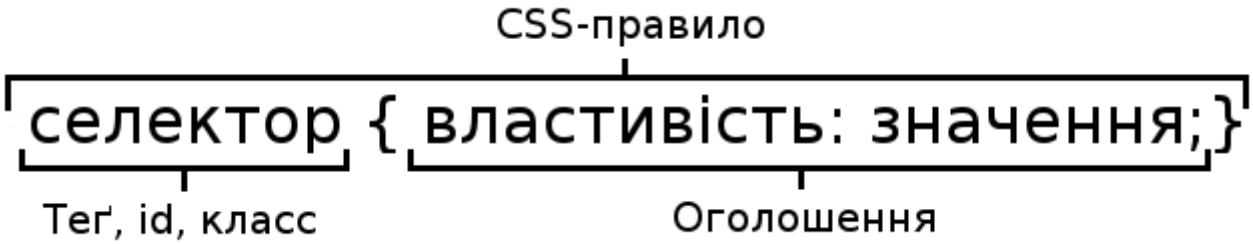
переміщення файлів
mv app1/*.* app2 - Перемістити всі файли з папки app1 в папку app2

Cascading Stylesheets
(CSS, каскадні таблиці стилів)
це код, що використовується для стилізації сайту. Це мова таблиць стилів, яка дозволяє задавати стилі обраним елементам у HTML документах



Вся структура називається "**rule set**" — "**набір правил**" (частіше, скорочено, "правило"). Його частини також мають свої назви:

- **Selector — Селектор** Назва елементу HTML на початку правила. Селектор вибирає елемент чи елементи, які будуть стилізовані (у нашому випадку, елементи p). Для стилізації інших елементів просто змініть селектор.
- **Declaration — Визначення (декларація)** Одне правило на зразок color: red; вказує, яку властивість елементу ви бажаєте стилізувати.
- **Properties — Властивості** Шляхи, якими ви можете стилізувати даний HTML елемент. (У цьому випадку, color — це властивість елементів p). У CSS ви обираєте властивості, які хочете змінити у вашому правилі.
- **Property value — Значення властивості** Праворуч від властивості, після двокрапки, ми вказуємо **значення властивості,** обираючи з багатьох можливих значень для наданої властивості (є багато інших значень властивості color окрім red).



Необхідно приєднати **CSS** до вашого **HTML** документу, інакше стилі **CSS** не вплинуть на те, як браузер буде відображати **HTML** документ.

```
<link rel="stylesheet" href="style.css">
```

Деякі CSS правила

color - колір тексту
background-color - фон
font-size - розмір шрифта
width - ширина
height – висота

Tables

Елемент <img> повинен містити файли у графічному форматі GIF, JPEG, SVG або PNG. Адреса файлу з картинкою задається через атрибут src.
GIF (Graphics Interchange Format — «формат обміну зображеннями»). Одними із головних особливостей формату є підтримка анімації та прозорості.
PNG (Portable Network Graphics) — збереження графічної інформації, що використовує стиснення без втрат. (Має прозорий фон).
JPEG (Joint Photographic Experts Group) – формат для зберігання фотозображень і подібних до них зображень.
SVG (Scalable Vector Graphics масштабована векторна графіка) — специфікація мови розмітки, що базується на XML, та формат файлів для двовимірної векторної графіки, як статичної, так і анімованої та інтерактивної.

Опис

Елемент **<table>** служить контейнером для елементів, що визначають вміст таблиці. Будь-яка таблиця складається з рядків і осередків, які задаються за допомогою тегів **<tr>** і **<td>** . Усередині **<table>** допустимо використовувати наступні елементи: **<caption>** , **<col>** , **<colgroup>** , **<tbody>** , **<td>** , **<tfoot>** , **<th>** , **<thead>** і **<tr>** .

Таблиці з невидимою кордоном довгий час використовувалися для верстки веб-сторінок, дозволяючи розділяти документ на модульні блоки. Подібний спосіб застосування таблиць знайшов втілення на багатьох сайтах, поки йому на зміну не прийшов більш сучасний спосіб верстки за допомогою шарів.

Синтаксис

```
<table>
  <tr>
    <td>...</td>
  </tr>
</table>
```

атрибути

align	Визначає вирівнювання таблиці.
background	Задає фоновий малюнок у таблиці.
bgcolor	Колір фону таблиці.
border	Товщина рамки в пікселях.
bordercolor	Колір рамки.
cellpadding	Відступ від рамки до вмісту осередку.
cellspacing	Відстань між осередками.
cols	Число колонок в таблиці.
frame	Повідомляє браузеру, як відображати межі навколо таблиці.
height	Висота таблиці.
rules	Повідомляє браузеру, де відображати кордону між осередками.
summary	Короткий опис таблиці.
width	Ширина таблиці.

Також для цього тега доступні універсальні атрибути і події .

закриває тег

Обов'язковий.

Таблиця має свою структуру

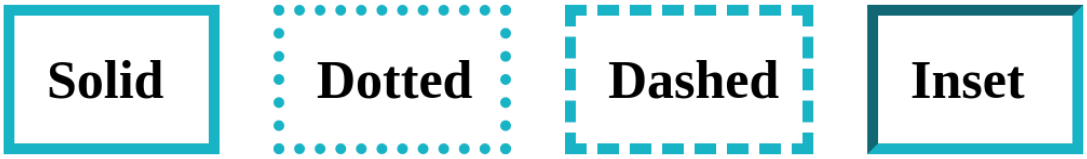
```
<!DOCTYPE html>
<html>
<body>
<h2>Basic HTML Table</h2>

<table style="width:30%">
  <tr>
    <th>Firstname</th>
    <th>Lastname</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
  </tr>
</table>
</body>
</html>
```

Basic HTML Table

Firstname	Lastname	Age
Jill	Smith	50
Eve	Jackson	94

border: 2px solid green;
2px - border width
solid - border type

green - border color



Класи

Як зробити однакові елементи різними?

Атрибут **class** задає стильовий клас, який дозволяє зв'язати певний тег зі стильовим оформленням.

HTML

```
<element class="classname">
```

CSS

```
.classname { color: red; }
```

ID - унікальне ім'я елемента, яке використовується для зміни його стилю і звернення до нього через скрипти.

HTML

```
<element id="idname">
```

CSS

```
#idname { color: red; }
```

GitHub

Як декілька людей розробляють один проект?

В Git для всего вычисляется хеш-сумма, и только потом происходит сохранение. В дальнейшем обращение к сохранённым объектам происходит по этой хеш-сумме. Это значит, что невозможно изменить содержимое файла или директории так, чтобы Git не узнал об этом. Данная функциональность встроена в Git на низком уровне и является неотъемлемой частью его философии. Вы не потеряете информацию во время её передачи и не получите повреждённый файл без ведома Git.

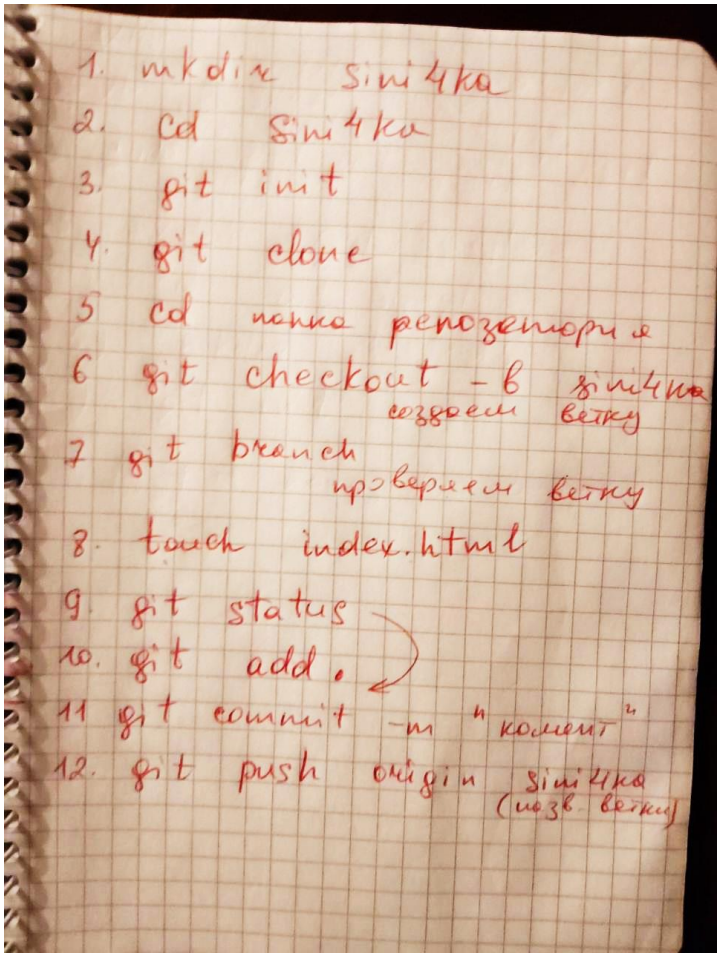
Механизм, которым пользуется Git при вычислении хеш-сумм, называется SHA-1 хеш. Это строка длинной в 40 шестнадцатеричных символов (0-9 и a-f), она вычисляется на основе содержимого файла или структуры каталога.

Git-директория — это то место, где Git хранит метаданные и базу объектов вашего проекта. Это самая важная часть Git, и это та часть, которая копируется при клонировании репозитория с другого компьютера. Рабочая директория является снимком версии проекта. Файлы распаковываются из сжатой базы данных в Git-директории и располагаются на диске, для того чтобы их можно было изменять и использовать.

Область подготовленных файлов — это файл, обычно располагающийся в вашей Git-директории, в нём содержится информация о том, какие изменения попадут в следующий коммит. Эту область ещё называют “индекс”, однако называть её stage-область также общепринято.

Власний акаунт GITHub

1. Go to https://github.com/
2. Click Sign Up
3. Verify your account using e-mail
4. Welcome to the **GIT family**



Блокова верстка сайту

Веб-сторінка складається з безлічі різних елементів, які можуть мати складну структуру. Тому при створенні веб-сторінки виникає необхідність потрібним чином позиціонувати ці елементи, стилізувати їх так, щоб вони розташовувалися на сторінці потрібним чином. Тобто виникає питання створення макета сторінки, її верстки.

Блокова верстка

це підхід, при якому сайт будують на основі блоків, як блоки виступають теги div.

Тег ** ** - це так званий контейнер (блок), який може містити форматований текст, зображення та ін. Важливою особливістю блоків є їх здатність накладатися один на одного при верстці. Ця особливість надає блокам набагато більше можливостей по верстці сайту, ніж, приміром, таблиці.

** - блоковий елемент, тому, якщо не задана ширина, розтягується на всю ширину вікна браузера**

DEFAULT

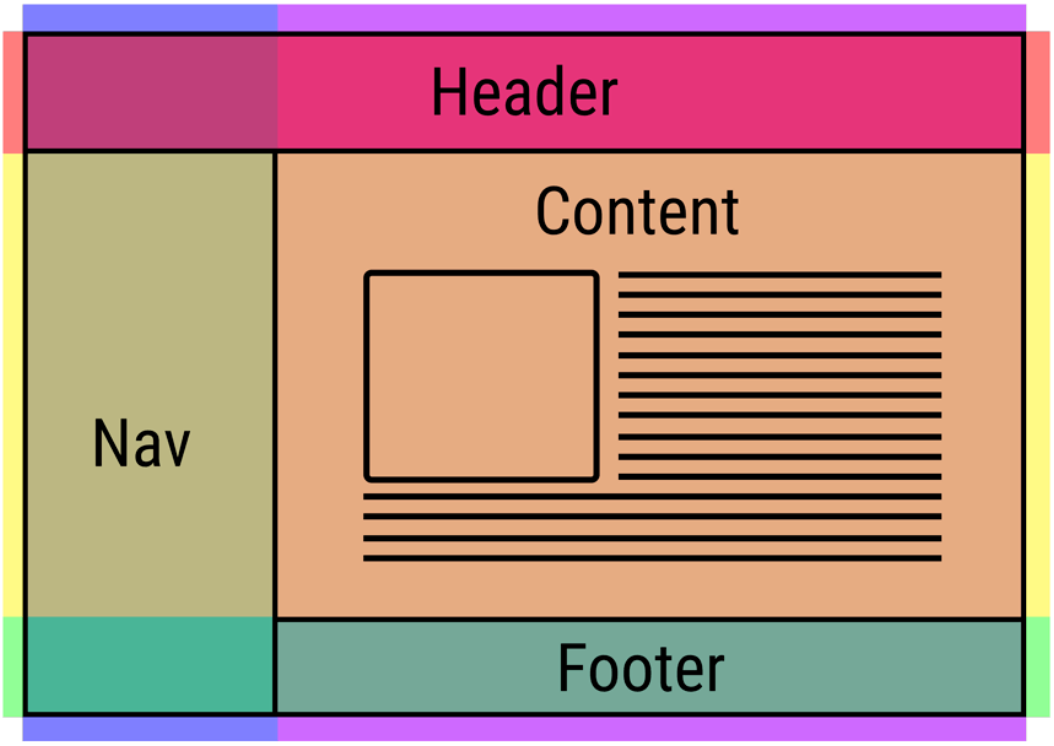
```
div {
width: 100%;
}
```

CUSTOM

```
div {
width: 300px;
}
```

** - не має оформлення. Щоб його побачити потрібно задати йому стилі в CSS. Висота блоку, коли її не вказано, дорівнює вмісту. Порожній блок div має висоту - 0 px, тому не відображається на сторінці**

```
div {
height: 300px;
}
```



Як же розташувати усі ці блоки?

Відображення елементів на сторінці браузера здійснюється в тому порядку, в якому вони слідують у HTML коді.

Але, це можна змінити за допомогою CSS. Стилі дають нам можливість позиціонувати вміст і елементи на сторінці. Існує декілька різних типів позиціонування в CSS

Властивість 'display'

Елементи, що мають блокове відображення (display: block) відображаються в потоці як прямокутні області, кожна з них на новій лінії одина під одній зверху вниз.

Елементи зі рядковим відображенням (display: inline) слідують один за одним зліва направо. Якщо простір праворуч закінчився, то вони переносяться на наступний рядок

Властивість 'float'

Визначає, з якого боку буде вирівнюватися елемент, при цьому інші елементи будуть обтікати його з інших сторін.

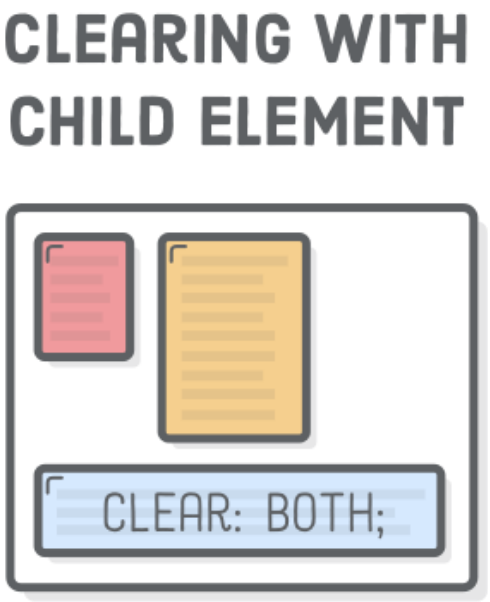
float: left | right | none | inherit

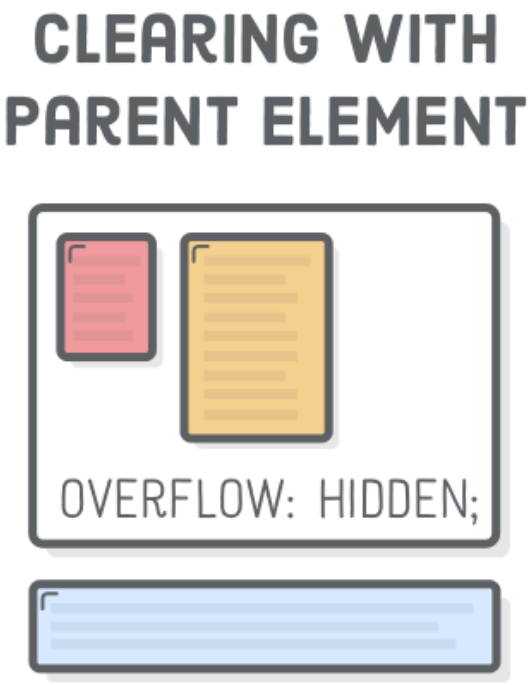
Властивість float має один побічний ефект. Її використання може впливати на батьківський елемент. Якщо такий елемент містить тільки float-елементи, то він нібито стискається, тобто його висота дорівнює нулю.

Як це виправити?

Метод overflow. Заснований на тому, що батьківського елементу необхідно встановити властивість overflow. Якщо значення цієї властивості встановлено в auto або hidden, то батьківський елемент збільшиться, щоб вмістити в себе всі float-елементи.

Властивість clear (left,right,both). Відміняє дію float. Будь-який елемент, у якого встановлено властивість clear, що не буде піднятий вгору, а відобразиться нижче, після float-елементів

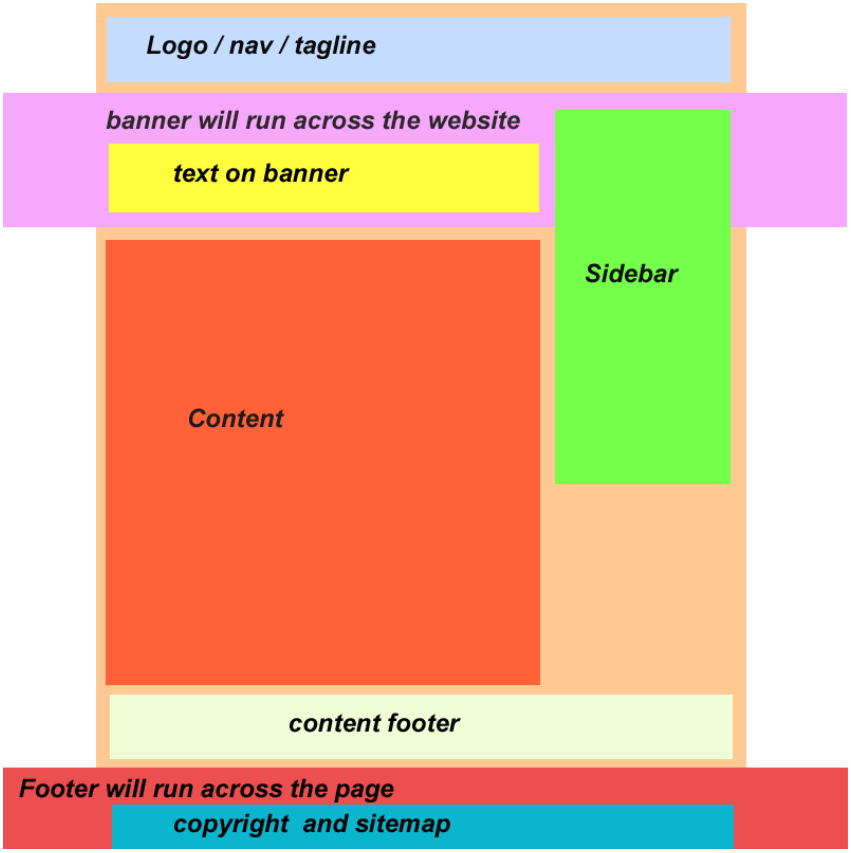




Position

А якщо потрібно розташувати блок “незвично”?

CSS властивість **position** - це одна з властивостей за допомогою якої можна змінити базову поведінку елементів в потоці. Іншими словами, ця властивість дозволяє «висмикнути» будь-який елемент з потоку документа і розмістити його в іншому місці щодо вікна браузера або інших елементів на веб-сторінці.



Яке буває позиціонування?

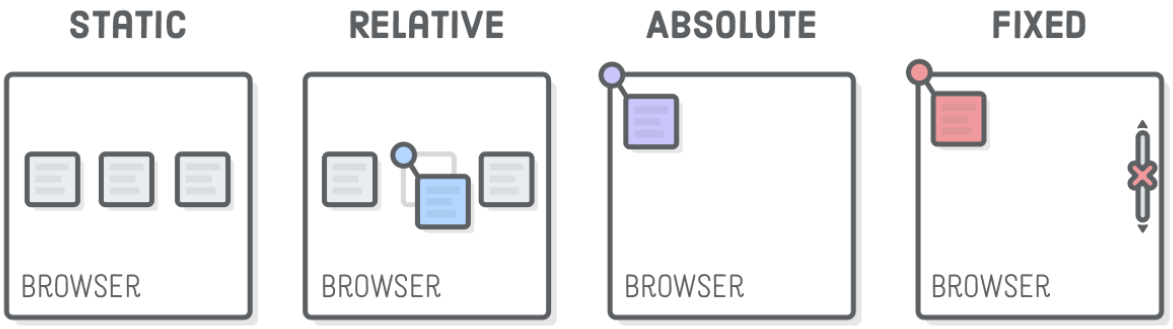
Властивість position має 5 значень:

- **static** (статичне позиціонування);
- **relative** (відносне);
- **absolute** (абсолютне);
- **fixed** (фіксоване);
- **sticky** (липке).

Воно застосовується в поєднанні з властивостями зміщення блоку, які точно визначають, де елемент буде розташований шляхом переміщення елемента в різних напрямках

Top (верх)
right (право)
bottom (низ)
left (ліво)

```
.offset {
  left: 20px;
  position: relative;
  top: 20px;
}
```



FlexBox

Як вирівнювати всі ці блоки?

CSS Flexible Box Layout Module, коротко flexbox (Флексбокс), створений щоб прибрати недоліки різних HTML конструкцій, в тому числі адаптованих під різну ширину і висоту,

Flexbox дозволяє контролювати найрізноманітніші параметри елементів контейнера: напрямок, порядок, ширину, висоту, вирівнювання вздовж і поперек, розподіл вільного місця, розтягнення і стиснення елементів.

Flexbox визначає набір CSS властивостей для контейнера (flex-контейнер) і його дочірніх елементів (flex-блоків). Перше, що потрібно зробити - це вказати контейнеру **display: flex**

Одним з основних понять в **flexbox** є осі.

Головною віссю flex-контейнера є напрям, відповідно до якого розташовуються всі його дочірні елементи.

Поперечною віссю називається напрямок, перпендикулярний головної осі.

Головна вісь за замовчуванням розташовується зліва направо. Поперечна - зверху вниз. Напрямок головної осі flex-контейнера можна задавати, використовуючи базове css властивість **flex-direction**.

Властивість **justify-content** визначає те, як будуть вирівняні елементи вздовж головної осі. Доступні значення **justify-content**:

- **flex-start**
- **flex-end**
- **center**
- **space-between**
- **space-around**
- **space-evenly**

Якщо justify-content працює з головною віссю, то **align-items** працює з віссю, перпендикулярної головній осі.

- **flex-start**
- **flex-end**
- **center**
- **stretch**
- **baseline**

flex-wrap: керує перенесенням елементів, що не поміщуються у контейнер .

- **nowrap (default)** - вкладені елементи розташовуються в один ряд/колонку незалежно від того поміщаються вони в контейнер чи ні.
- **wrap** - включає перенесення елементів на наступний ряд, якщо вони не поміщаються в контейнер.
- **wrap-reverse** - теж що wrap тільки перенесення буде в зворотному напрямку).

https://codepen.io/enxaneta/full/adLPwv - на допомогу кожному