projektowanie stron WWW

Magdalena Madej

kontakt: mmadej@wsb.gda.pl

przygotowanie środowiska pracy

edytory tekstu

VISUAL STUDIO CODE

https://code.visualstudio.com/

SUBLIME TEXT

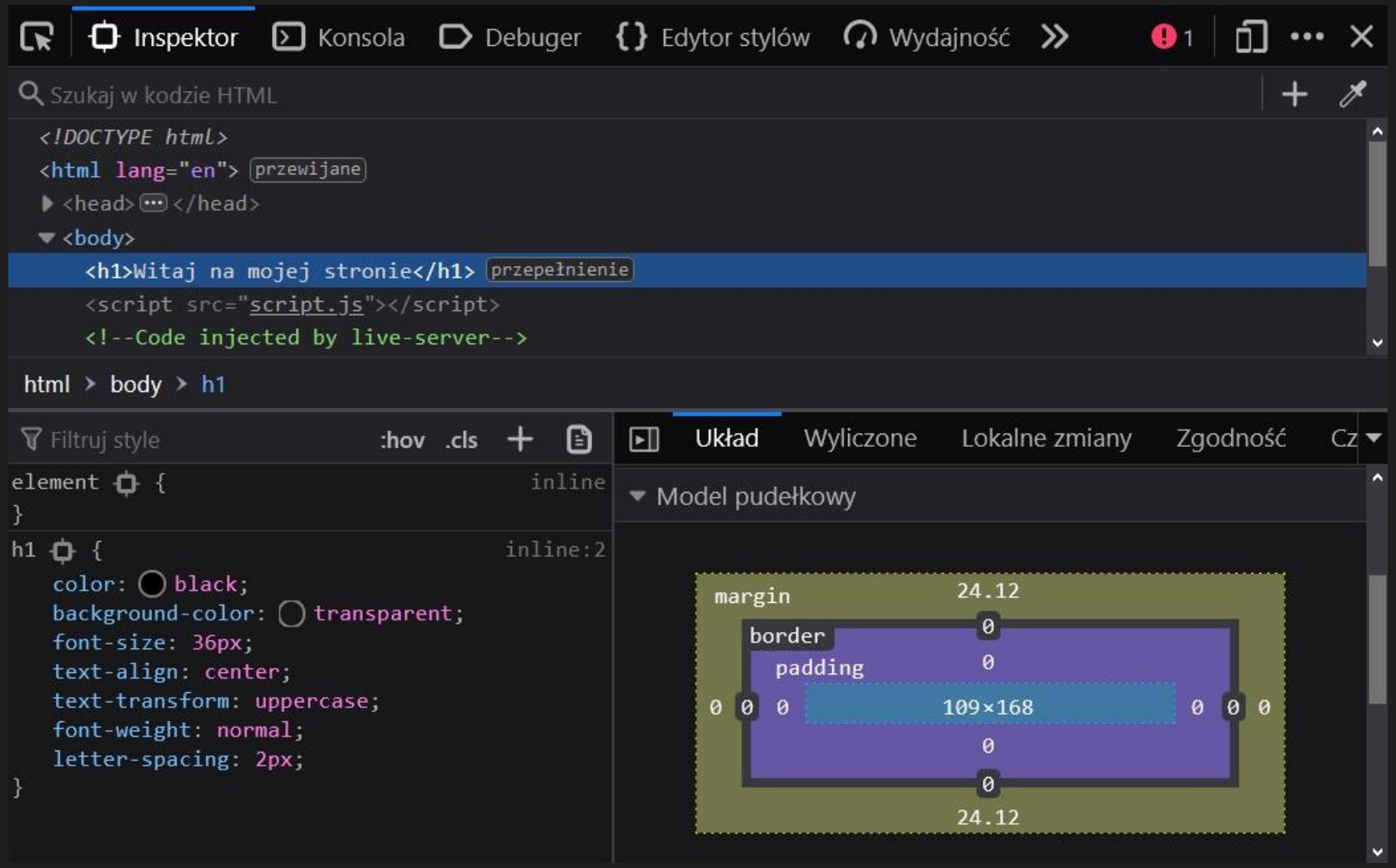
https://www.sublimetext.com/

BRACKETS

http://brackts.io/

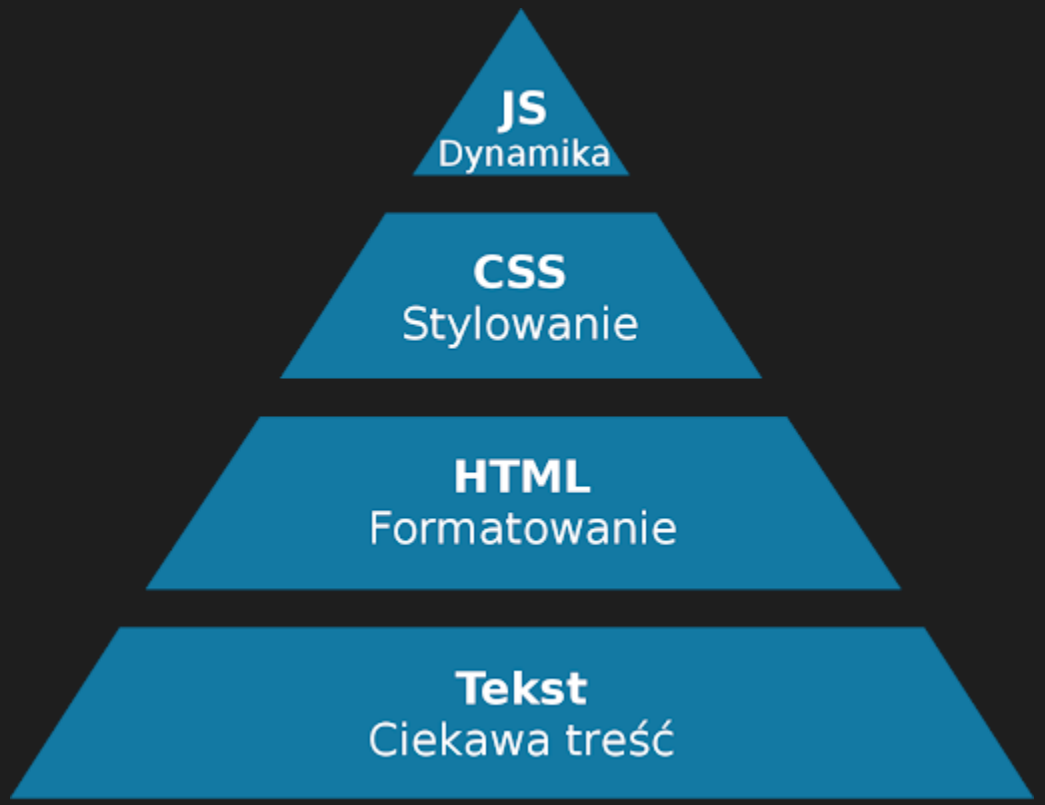
https://hackr.io/blog/web-development-ide

narzędzia programistyczne w przeglądarce



zagadnienia wprowadzające

warstwy strony internetowej



JavaScript – odpowiada za dynamikę stron www, zarówno w postaci pojawiania się i przemieszczania elementów, jak i interakcji z użytkownikiem

CSS – kaskadowe arkusze stylów – służą do prezentacji i rozmieszczenia elementów na stronie oraz odpowiadają za prostą dynamikę

HTML – odpowiada za semantykę i formatowanie treści

tekst – suche dane, które chcemy przedstawić na stronie

rodzaje stron internetowych

wizytówka

rodzaj strony internetowej, która przekazuje podstawowe, a zarazem najważniejsze informacje. Zazwyczaj wizytówki mają formę One Page.

strona firmowa

taka strona zazwyczaj składa się z wielu zakładek, gdzie możesz znaleźć szczegółowe i przydatne informacje, np. kontakt, ofertę, poprzednie realizacje itp.

portal internetowy

portale internetowe służą przede wszystkim do przekazywania informacji. Są rozbudowane i muszą być przygotowane na masowy ruch i dużą ilość treści.

wortal

wyspecjalizowana strona internetowa, która jest poświęcona tylko jednej tematyce np. fotografii, kina, muzyki itp.

blog

najważniejszym celem bloga jest stworzenie lojalnej społeczności. Umożliwia swobodne pisanie porad, informacji itp.

sklep internetowy

sklep internetowy można wykorzystywać przede wszystkim do sprzedawania produktów bądź usług. Jego najważniejszym celem jest zaprezentowanie oferty.

HTML

wprowadzenie

- HTML - Hypertext Markup Language
- pozwala opisać strukturę informacji zawartych wewnątrz strony internetowej, nadając odpowiednie znaczenie semantyczne poszczególnym fragmentom tekstu
- nie jest językiem programowania

https://www.w3.org/standards/webdesign/htmlcss

budowa tagów (znaczników)

tagi i atrybuty

atrybuty zawierają dodatkowe informacje o elementach. Składają się z dwóch części: nazwy oraz wartości

atrybuty globalne

atrybuty globalne mogą być używane w dowolnym elemencie HTML

atrybut	**wartości**	**opis**
class	ciąg tekstowy	Przypisuje elementowi jedną lub więcej klas. Jeśli wartości jest wiele, należy rozdzielić je spacjami
id	ciąg tekstowy	Przypisuje elementowi unikatowy identyfikator
style	rozdzielona średnikami lista reguł CSS	Dodaje elementowi informacje dotyczące stylów CSS
title	ciąg tekstowy	Zawiera tytuł lub informacje pomocnicze dotyczące elementu, zwykle wyświetlane jako podpowiedź

struktura

```
<!DOCTYPE html>
<html lang="pl">

        <head>
                <meta charset="UTF-8">
                <title>Document</title>
                <!-- sekcja zawierająca informacje dla przeglądarek. Użytkownik nie widzi sekcji head, poza
tagiem <title> -->

        </head>

        <body>
                <!-- sekcja, którą widzi użytkownik w oknie przeglądarki -->
        </body>

</html>
```

metatagi

metatagi są umieszczone wewnątrz elementu <head> i zawierają informacje dotyczące stron www.

```
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta name="keywords" content="">
<meta name="description" content="">
```

semantyka

istnieje grupa znaczników, których przeznaczeniem nie jest określanie jej struktury, lecz nadawanie jej dodatkowego znaczenia. Są to znaczniki semantyczne

semantyka dla deweloperów

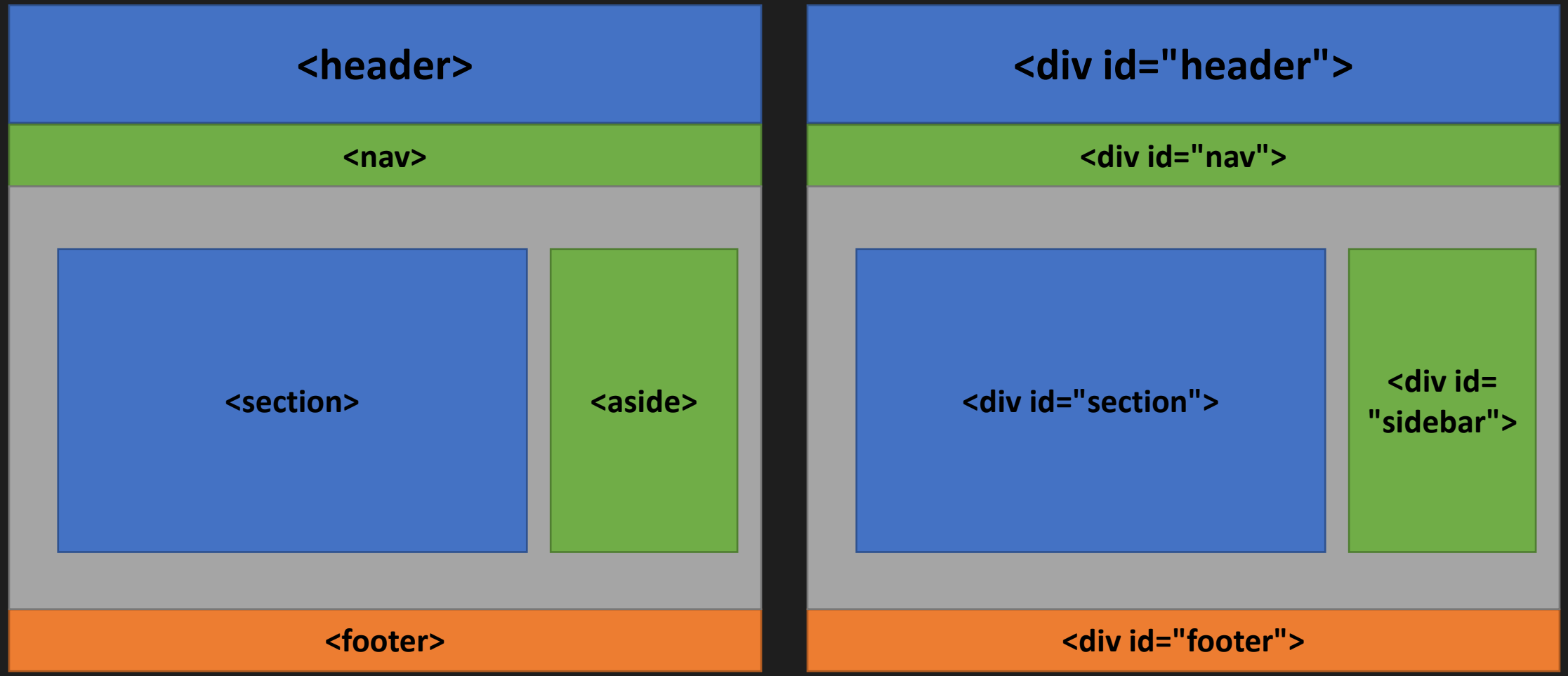
- pozwala pisać czysty i przejrzysty kod
- takie same zasady obowiązują wszystkich programistów, co znacznie ułatwia pracę
- obowiązuje jeden standard pisania kodu
- strona jest wyżej notowana w Google

semantyka dla osób mających problemy ze wzrokiem

- stosując się do semantyki ułatwiamy korzystanie ze stron osobom, które mają problemy ze wzrokiem i korzystają z czytników ekranowych

https://www.youtube.com/watch?v=dEbl5jvLKGQ

nowe tagi semantyczne



elementy blokowe i liniowe

elementy blokowe

- zaczynają się od nowej linii
- zajmują 100% szerokości rodzica
- mogą zawierać zarówno elementy liniowe jak i inne elementy blokowe

```
<p>tekst w paragrafie</p>
```

- p, div
- h1, h2, h3, h4, h5, h6
- ol, ul
- table

elementy liniowe

- nie tworzą nowego wiersza
- zajmują tyle miejsca ile potrzebują
- mogą zawierać tylko tekst lub inne elementy liniowe

```
<span>tekst w spanie</span>
```

- b, big, i, small,
- abbr, acronym, cite, code, dfn, em, strong,
- a, br, img, script, span, sub, sup
- button, input, label, textarea

tekst

nagłówki

- HTML udostępnia sześć „poziomów” nagłówków.
- najważniejszy na stronie jest nagłówek <h1>

```
<h1>nagłówek pierwszego poziomu</h1>
<h2>nagłówek drugiego poziomu</h2>
<h3>nagłówek trzeciego poziomu</h3>

<h6>nagłówek szóstego poziomu</h6>
```

akapity

- tekst w HTML należy umieszczać wewnątrz tagów

```
<p>Lorem ipsum, dolor sit amet consect etur adipisicing elit. Officiis, animi.</p>
```

edycja tekstu

pogrubienie tekstu

pogrubiony tekst - zmiana wyglądu tekstu i podniesienie wartości semantycznej

pogrubiony tekst - zmiana wyglądu tekstu

pochylenie tekstu

tekst pochylony - zmiana wyglądu tekstu i podniesienie wartości semantycznej

tekst pochylony - zmiana wyglądu tekstu

przekreślenie tekstu

~~przekreślony tekst~~ - usunięcie tekstu, ale równocześnie pozostawienie na stronie

<s>przekreślony tekst</s> - prezentacja treści nieaktualnej

podkreślenie tekstu

<u>tekst podkreślony</u> - zmiana wyglądu tekstu

tekst

indeks górny

element <sup> jest stosowany do oznaczenia znaków, które powinny zostać wyświetlone jako indeks górny

```
<p>E = mc<sup>2</sup></p>
```

indeks dolny

element <sub> jest stosowany do oznaczenia znaków, które powinny zostać wyświetlone jako indeks dolny

```
<p>CO<sub>2</sub></p>
```

podział wiersza

- aby zakończyć wiersz i wyświetlić dalszą część akapitu w nowym, można to zrobić używając znacznika

```
<p>Lorem ipsum dolor sit<br />amet consectetur adip
isicing elit.</p>
```

- innym sposobem, aby uzyskać ten sam efekt jest zapianie tekstu w znaczniku <pre>

```
<pre>
    imię i nazwisko
    adres mailowy
</pre>
```

obrazy

aby umieścić obraz na stronie www, należy użyć znacznika <img>. Jest to element, który nie posiada znacznika zamykającego.

- atrybut alt zawiera tekst stanowiący opis obrazu, z którego można skorzystać, gdy obraz będzie niewidoczny

zasady tworzenia obrazów

1

ZAPISUJ OBRAZY W ODPOWIEDNIM FORMACIE

w witrynach www zazwyczaj używa się obrazów zapisanych w formatach: .jpeg, .gif, .png.

2

ZAPISUJ OBRAZY O ODPOWIEDNICH WYMIARACH

zapisywane obrazy powinny mieć takie wymiary jak te, które będą widoczne na stronie.

3

UŻYWAJ ODPOWIEDNIEJ ROZDZIELCZOŚCI

tworzenie obrazów o dużej rozdzielczości sprawia, że zwiększa się czas wczytywania strony.

listy

uporządkowana

```
<ol>
        <li>pierwszy element</li>
        <li>drugi element</li>
        <li>trzeci element</li>
</ol>
```

nieuporządkowana

```
<ul>
        <li>pierwszy element</li>
        <li>drugi element</li>
        <li>trzeci element</li>
</ul>
```

- sprawdzić: `<li type="a"></li>`

lista definicji

listę definicji tworzy się przy użyciu elementu <dl>. Zwykle zawiera serię terminów wraz z ich opisami

```
<dl>
    <dt>hasło</dt>
    <dd>definicja</dd>

    <dt>hasło</dt>
    <dd>definicja</dd>
</dl>
```

łącza

łącza tworzy się przy użyciu elementu **<a>.** Strona docelowa, z którą chcemy się połączyć, jest określona za pomocą atrybutu **href**.

```
<a href="http://www.google.com/">Google</a>
```

łącza do innych witryn

```
<ul>
        <li>
        <a href="google.pl/">
        Google</a></li>

        <li>
        <a href="youtube.com/">
        YouTube</a></li>
</ul>
```

łącza do stron własnej witryny

```
<ul>
        <li><a href="index.html">
         strona główna </a></li>
        <li><a href="about.html">
         o nas </a></li>
        <li><a href="contact.html">
        kontakt</a></li>
</ul>
```

otwieranie stron w nowym oknie przeglądarki

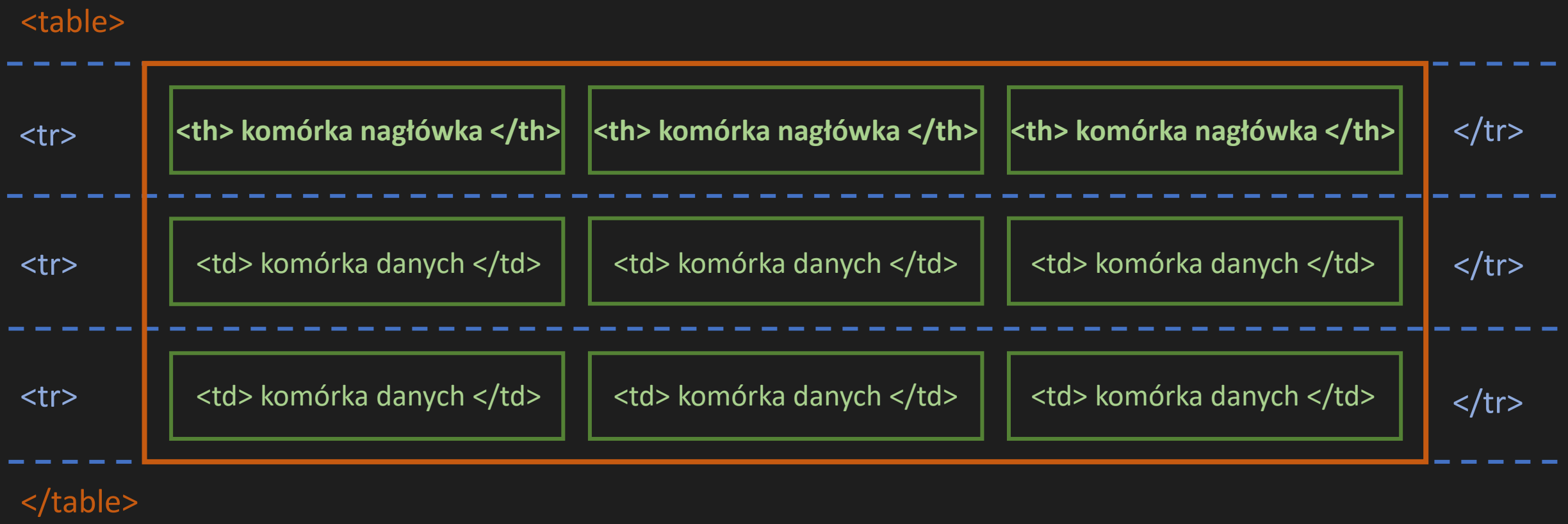
ustawienie atrybutu target="_blank" sprawia, że przeglądarka otworzy kliknięty link w nowej karcie

```
<a href="http://google.com" target="_blank">Google</a>
```

alternatywne typy linków

`<a href="mailto:mail@mail.com">Napisz do mnie maila</a>`	link pozwalający uruchomić aplikację do wysyłania poczty e-mail
`<a href="tel:+48123456789">Zadzwoń do mnie</a>`	link pozwalający wykonać połączenie telefoniczne
`<a href="callto:USERNAME">Zadzwoń na Skype do USERNAME</a>`	link pozwalający wykonać połączenie w aplikacji Skype
`<a href="sms:+48987654321">Wyślij do mnie SMS</a>`	link pozwalający napisać wiadomość SMS

tabele


<table>
<tr>
<th> komórka nagłówka </th>
<th> komórka nagłówka </th>
<th> komórka nagłówka </th>
</tr>
<tr>
<td> komórka danych </td>
<td> komórka danych </td>
<td> komórka danych </td>
</tr>
<tr>
<td> komórka danych </td>
<td> komórka danych </td>
<td> komórka danych </td>
</tr>
</table>

tabele

```
<table>
    <caption>Tytuł tabeli</caption>
    <thead>
        <tr>
            <th></th>
        </tr>
    </thead>
    <tbody>
        <tr>
            <td></td>
        </tr>
    </tbody>
</table>
```

- `<table>` — tworzenie tabeli
- `<thead>` — nagłówek tabeli
- `<tr>` — początek nowego wiersza
- `<th></th>` — komórka nagłówka tabeli
- `<tbody>` — główna zawartość tabeli
- `<td></td>` — komórka tabeli

elementy grup wierszy tabeli

<thead>...</thead>
grupa wiersza nagłówka tabeli

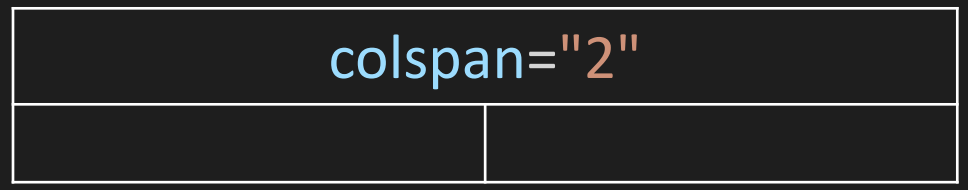
<tbody>...</tbody>
grupa wiersza ciała tabeli

<tfoot>...</tfoot>
grupa wiersza stopki tabeli

tabele

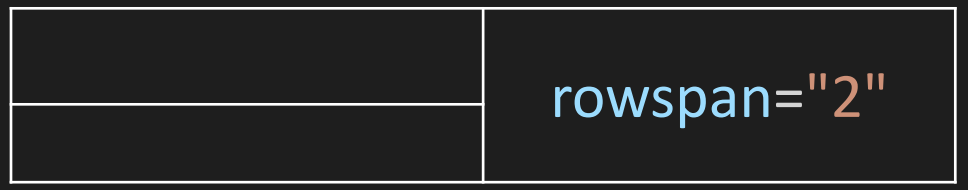
łączenie komórek z sąsiadujących kolumn

```
<td colspan="2"></td>
```



łączenie komórek w sąsiadujących wierszach

```
<td rowspan="2"></td>
```



formularze

formularze są dodawane do stron internetowych na pomocą elementu <form>, który jest pojemnikiem dla wszystkich elementów składowych formularza

```
<form>
    <!--tutaj pojawiają się elementy formularza -->
</form>
```

formularze

atrybut action

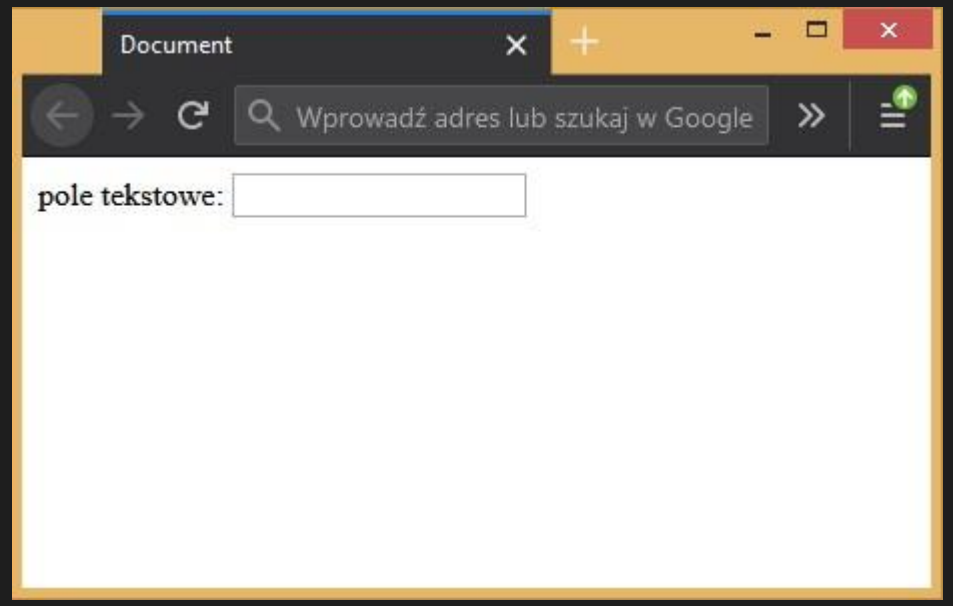
atrybut action wskazuje lokalizację (adres URL) aplikacji lub skryptu, które zostaną wykorzystane do przetworzenia formularza

```
<form action="adres URL">…</form>
```

kontrolki do wprowadzania tekstu

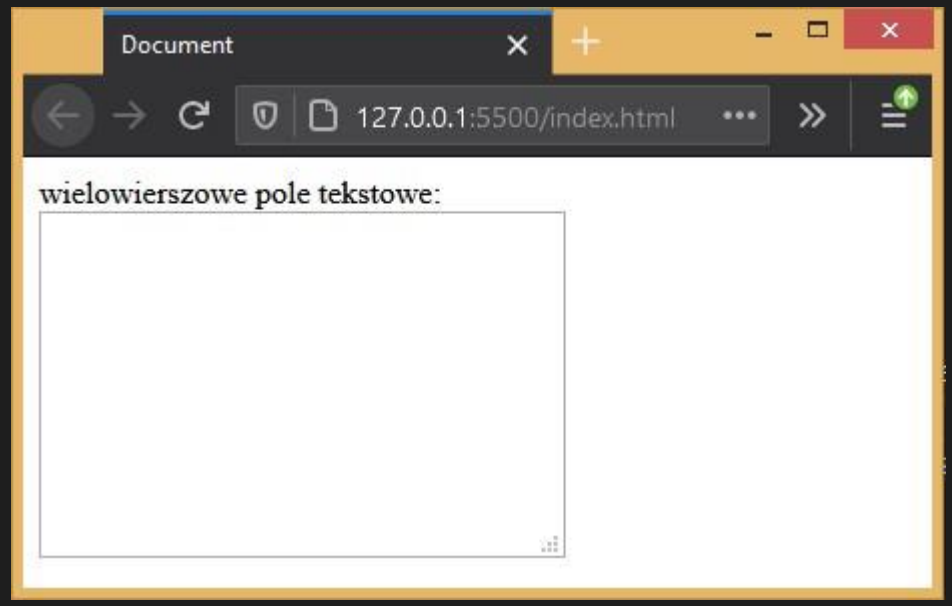
Jednowierszowe pole tekstowe

```
<label for="">pole tekstowe:</label>
<input type="text" name="" id="">
```



Wielowierszowe pole tekstowe

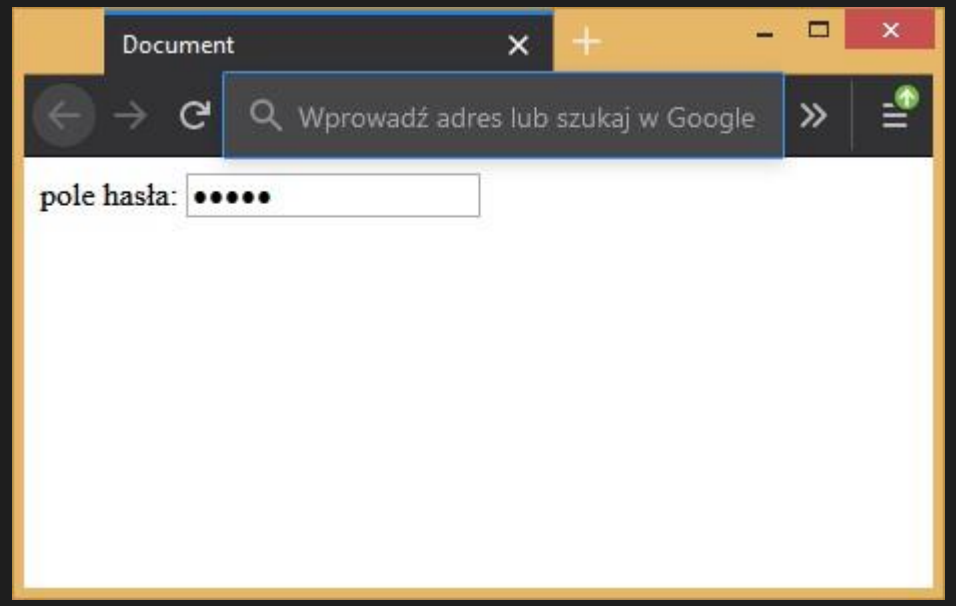
```
<textarea name="" id="" cols="30"
rows="10"></textarea>
```



wyspecjalizowane pola tekstowe

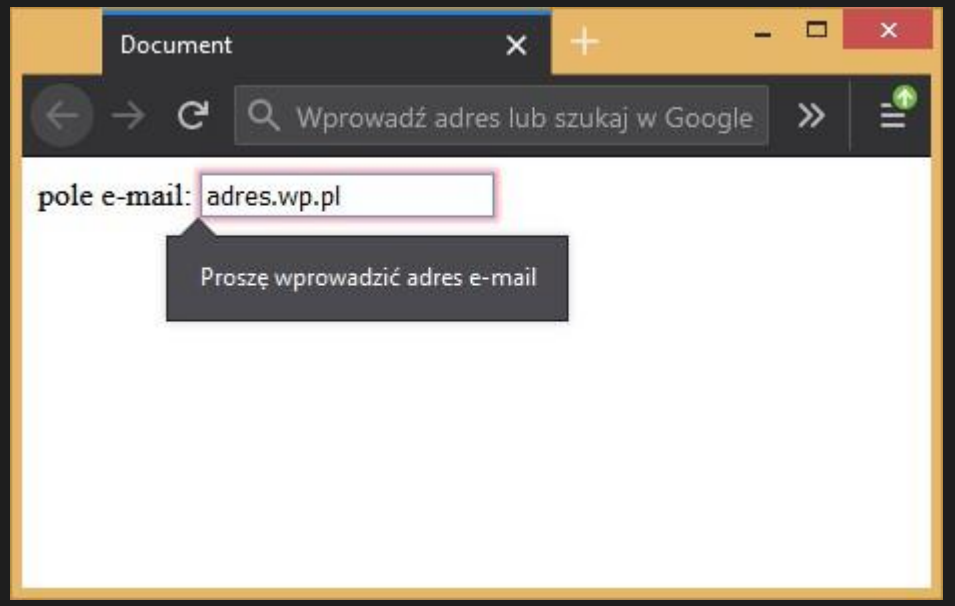
Pole hasła

```
<label for="">pole hasła:</label>
<input type="password" name="">
```



Pole adresu mailowego

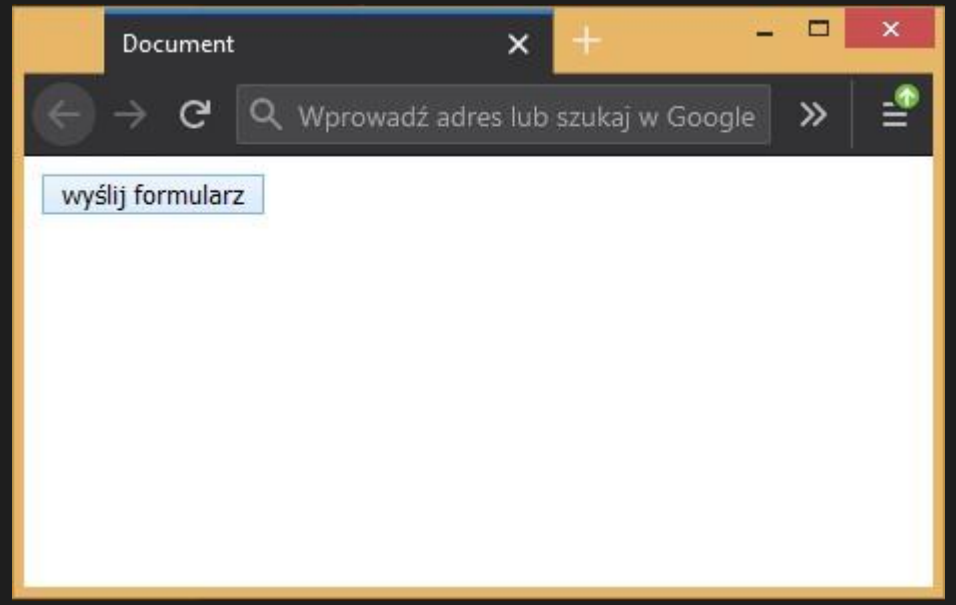
```
<label for="">pole e-mail:</label>
<input type="email" name="" id="">
```



przyciski submit i reset

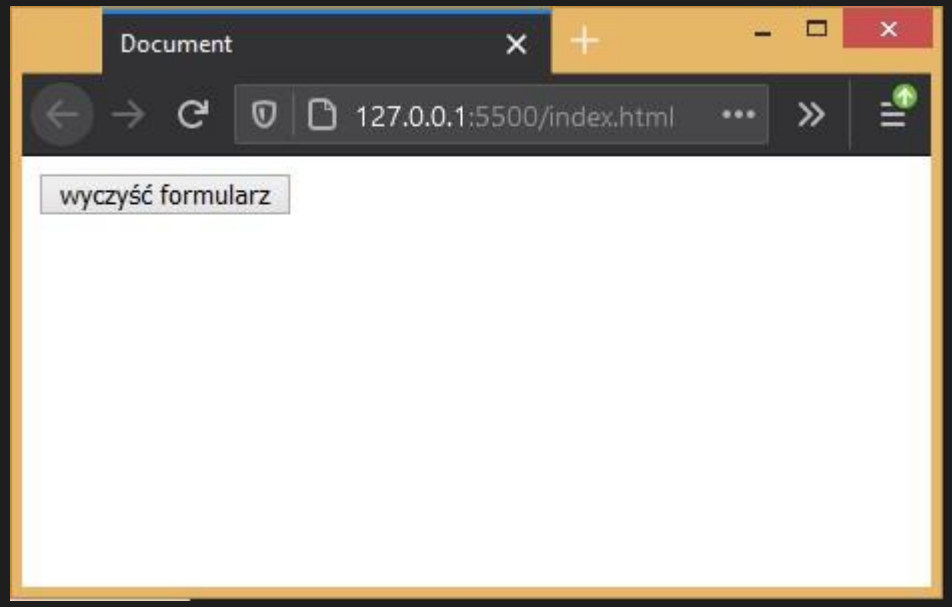
Przycisk submit

```
<input type="submit"
value="wyślij formularz">
```



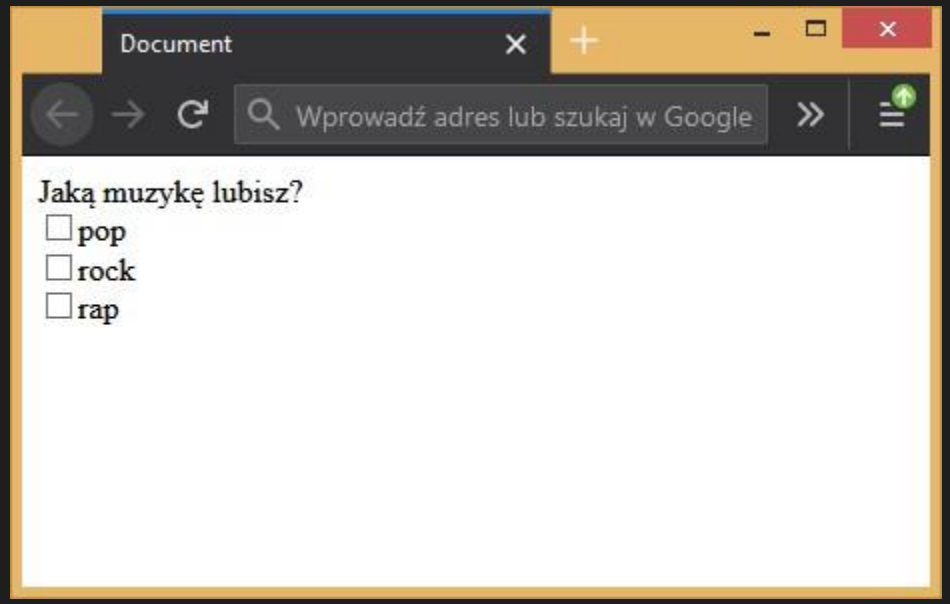
Przycisk reset

```
<input type="reset" name="" id=""
value="wyczyść formularz">
```



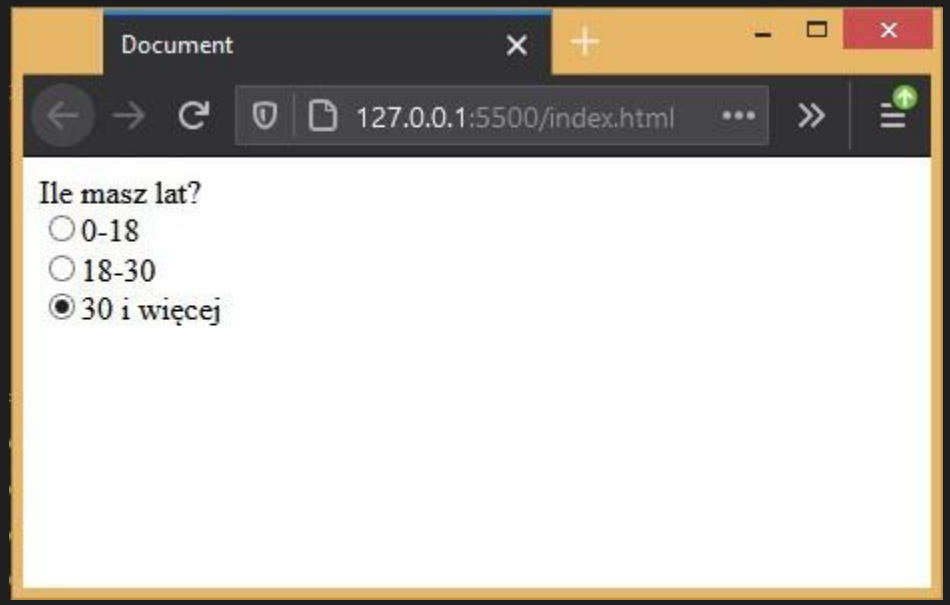
przycisk opcji – checkbox

```
<p>Jaką muzykę lubisz?</p>

<label for=""><input type="checkbox"
name="music">pop</label>
<label for=""><input type="checkbox"
name="music">rock</label>
<label for=""><input type="checkbox"
name="music">rap</label>
```



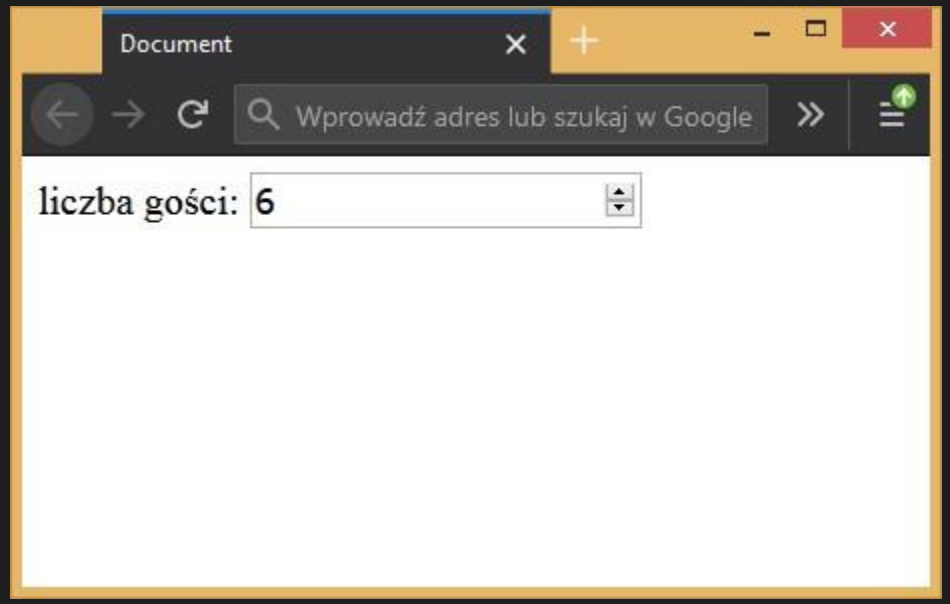
przycisk opcji – radio button

```
<p>Ile masz lat?</p>

<label for=""><input type="radio"
name="age">0-18</label>
<label for=""><input type="radio"
name="age">18-30</label>
<label for=""><input type="radio"
name="age">30 i więcej</label>
```



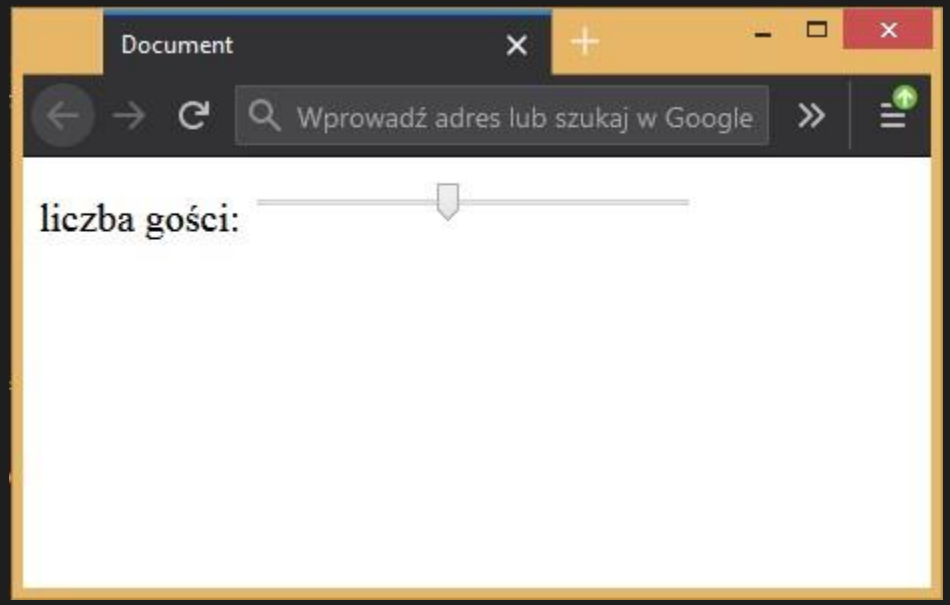
pole numeryczne

```
<label>
liczba gości: <input type="number"
name="guests" min="1" max="10">
</label>
```



pole numeryczne – suwak

```
<label>
liczba gości: <input type="range"
name="guests" min="1" max="10">
</label>
```



formularze – etykiety

element label jest wykorzystywany do powiązania opisowego tekstu z odpowiadającym mu polem formularza

każdy element label jest powiązany z dokładnie jedną kontrolką formularza. Można go użyć na 2 sposoby:

- powiązanie niejawne
- powiązanie jawne

etykiety – powiązanie niejawne

powiązanie niejawne to zagnieżdżenie kontrolki wraz z opisem wewnątrz elementu label

```
<label>
        liczba gości: <input type="number" name="guests" min="1"
        max="10">
</label>
```

etykiety – powiązanie jawne

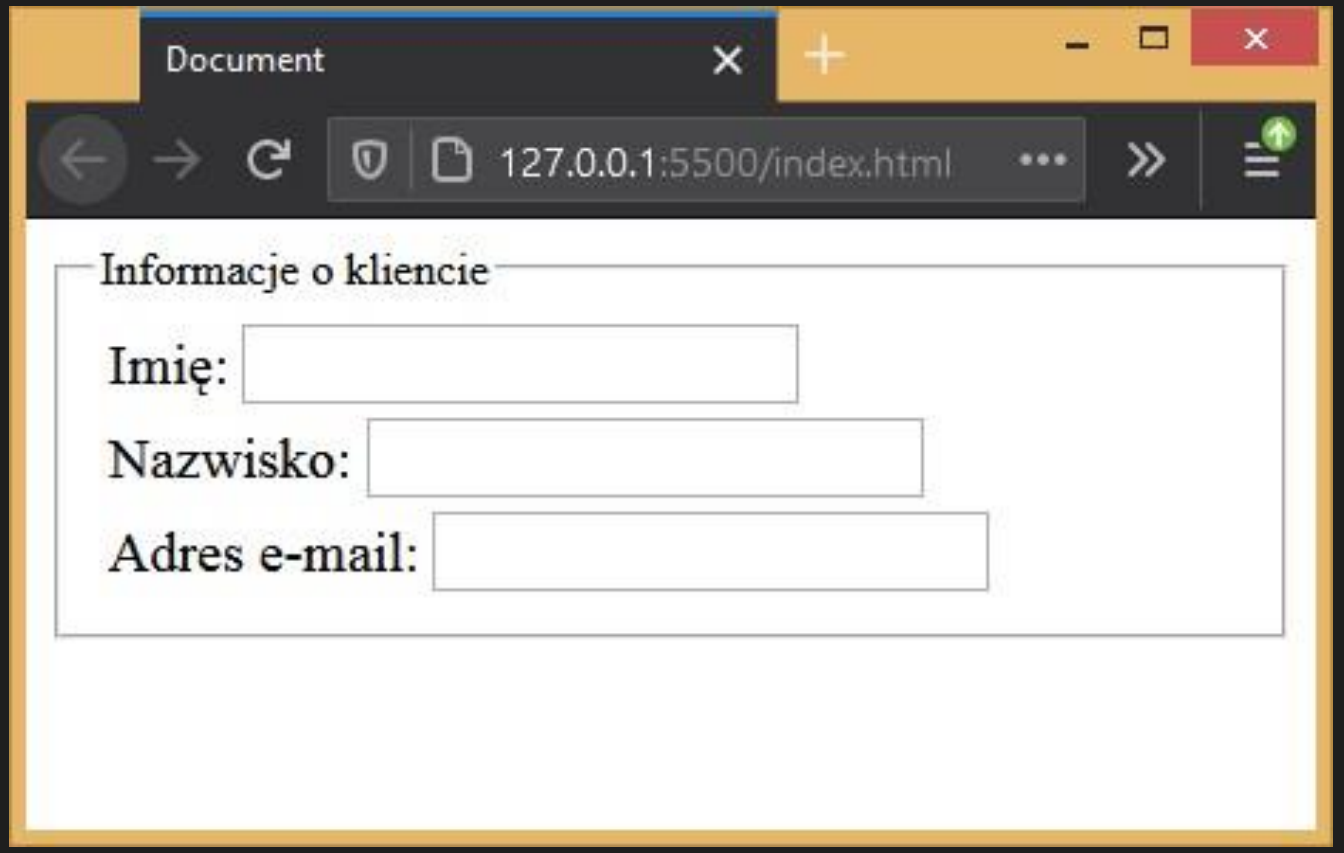
powiązanie jawne dopasowuje etykietę do identyfikatora (id) kontrolki. Atrybut for określa, dla jakiej kontrolki jest to etykieta.

```
<label for="username">Nazwa użytkownika:</label>
<input type="text" id="username">
```

formularze – grupowanie elementów

```
<fieldset>
        <legend>Informacje o kliencie</legend>
        <label>Imię: <input type="text"></label>
        <label>Nazwisko: <input type="text"></label>
        <label>Adres email: <input type="email"></label>
</fieldset>
```

formularze – grupowanie elementów



audio i video

umieszczanie plików audio

```
<audio src="plik.mp3"
controls="controls"></audio>
```

inne atrybuty:

- loop – odtwarzanie w pętli
- autoplay – automatyczne rozpoczęcie odtwarzania

umieszczanie plików video

```
<video src="plik.mp4"
controls="controls"></video>
```

Jak umieścić film z YouTube?

dodatkowe elementy HTML

symbole specjalne

¢ znak centa
¢
¢

© znak praw autorskich
©
©

× znak mnożenia
×
×

€ znak euro
€
€

™ znak handlowy
™
™

÷ znak dzielenia
÷
÷

£ znak funta
£
£

pozostałe elementy HTML

lista elementów dostępnych w HTML5 dostępna jest pod linkiem:

https://developer.mozilla.org/pl/docs/Web/HTML/Element

gdzie szukać informacji

- https://developer.mozilla.org/pl/
- https://devdocs.io/
- https://www.w3.org/ (walidator)

- https://crossweb.pl/

CSS

wprowadzenie

- CSS - Cascading Style Sheet
- kod używany do prezentacji i rozmieszczenia elementów na stronie, oraz odpowiedzialny za prostą dynamikę
- działanie kaskadowych arkuszy stylów polega na kojarzeniu reguł z elementami HTML

budowa reguł

reguły określają w jaki sposób ma być wyświetlana zawartość konkretnych elementów. Składają się z dwóch części: selektora i deklaracji

budowa reguł

każda deklaracja składa się z dwóch części: właściwości i wartości, oddzielonych od siebie znakiem dwukropka

```
h1, h2, h3 {
        color: royalblue;
        padding: 20px;
}
```

właściwość wartość

właściwość wskazuje konkretny aspekt wyglądu elementu, który chcemy określić

wartość wskazuje konkretne ustawienia, które chcemy nadać wybranej właściwości

umieszczanie stylów na stronie

- stosowanie zewnętrznych arkuszy stylów

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

- znacznik <style>

```
<style>
    p {color: royalblue;}
</style>
```

- style liniowe (stary zapis przed HTML5)

```
<p style="color: red;">tekst w paragrafie.</p>
```

selektory

selektor	znaczenie	przykład
selektor uniwersalny	odnosi się do wszystkich elementów strony	* { } odnosi się do wszystkich elementów na stronie
selektor typu	odnosi się do elementów o konkretnej nazwie	h1, h2, h3 { } odnosi się do elementów <h1>, <h2>, <h3>

selektory

selektor	znaczenie	przykład
selektor elementu dziecka	odnosi się do elementów, które są bezpośrednimi dziećmi innego, określonego elementu	li > a { } odnosi się do wszystkich elementów <a> umieszczonych bezpośrednio w elementach
selektor elementu potomnego	odnosi się do elementów będących elementami potomnymi innego, określonego elementu	section span { } odnosi się do dowolnego elementu umieszczonego wewnątrz elementu <section>, nawet jeśli znajduje się on wewnątrz jakichś innych elementów
selektor elementów sąsiadujących bezpośrednio	odnosi się do elementu bezpośrednio sąsiadującego z innym, określonym elementem	h1 + p { } odnosi się do pierwszego elementu <p> umieszczonego za elementem <h1> (lecz nie do kolejnych elementów <p>)
ogólny selektor elementów sąsiadujących	odnosi się do elementu sąsiadującego z innym elementem, choć niekoniecznie bezpośrednio	h1 ~ p { } jeśli na stronie są dwa elementy <p> sąsiadujące z elementem <h1>, to reguła odnosi się do obu z nich

class i id

class

- tę samą klasę możemy dodać do kilku elementów HTML
- w CSS odwołujemy się do klas za pomocą kropki
- możemy umieszczać kilka klas w jednym elemencie
- najlepiej, żeby nazwa była po angielsku i opisywała to, do czego służy

```
<p class="red top">tekst</p>

.red {
    color: red;
}
```

id

- id jest unikalne i może pojawić się tylko jedno na stronie
- w CSS odwołujemy się za pomocą hasha

```
<button id="basic">START</button>

#basic {
    text-align: center;
}
```

kaskadowość

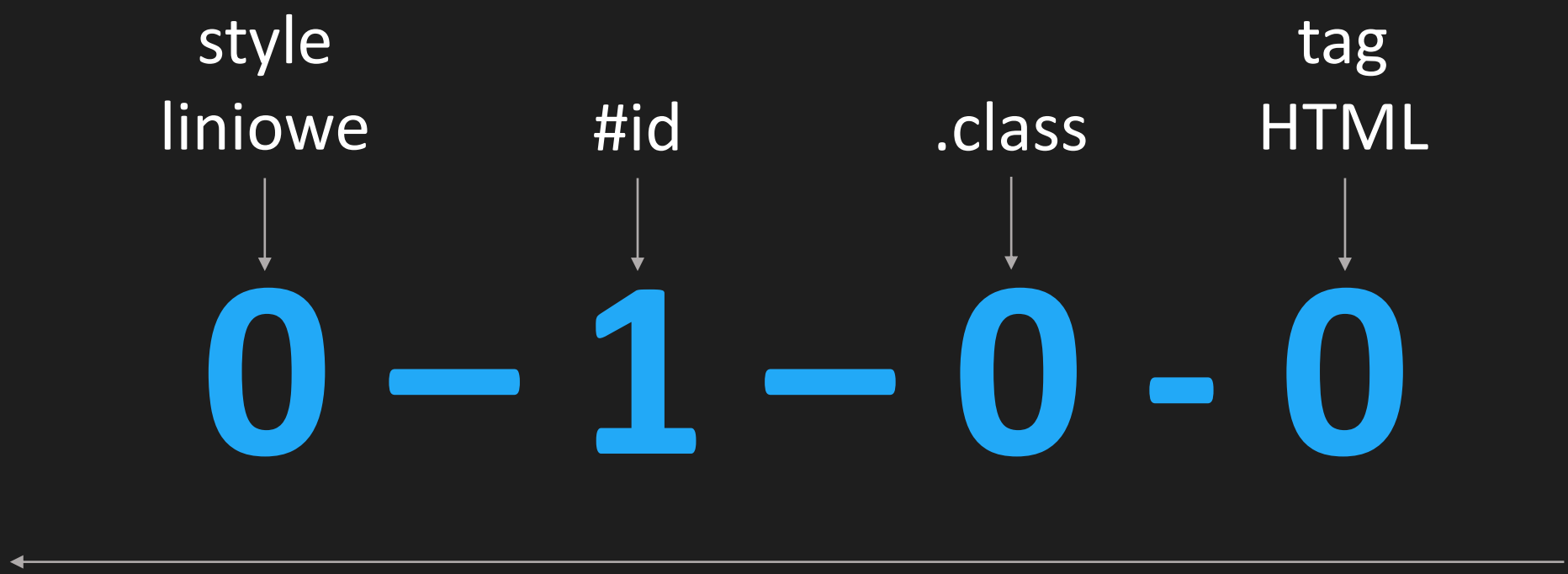
kaskadowość – to zasady określające, w jaki sposób przeglądarka wybiera wartość właściwości CSS dla danego elementu.

Jak poprawnie umieszczać style CSS

style przeglądarki < style zewnętrzne < nasze style CSS < znacznik <style>

przeglądarka wczytuje kod od góry do dołu

specyficzność



dziedziczenie

istnieją takie właściwości CSS, które są dziedziczone przez elementy potomne (np. font-family lub color). Są też takie, na których można wymusić dziedziczenie (np. padding). W tym celu wystarczy wpisać takiej właściwości wartość *inherit.*

rodzic – dziecko

```
<div>
    <p>dziecko div</p>
</div>
```

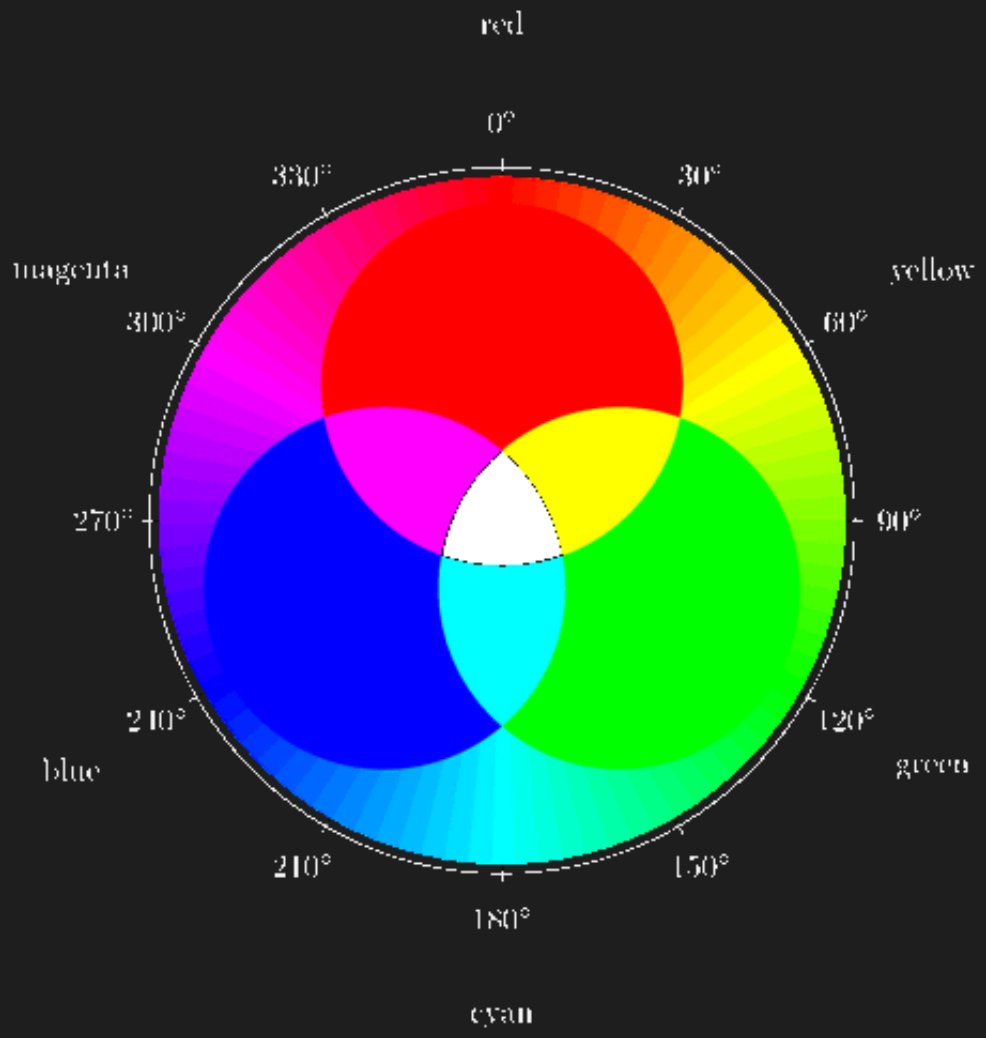
rodzeństwo

```
<div></div>
<div></div>
<div></div>
```

elementy potomne

```
<main>
    <section>
        <div>
            <p></p>
        </div>
    </section>
</main>
```

kolory



Kiedy monitor jest wyłączony, jest czarny, gdyż piksele nie emitują żądnego światła. Po włączeniu każdy piksel może mieć inny kolor.

Kolory na ekranie są tworzone jako połączenie pewnych ilości trzech kolorów podstawowych: czerwonego (r), zielonego (g) i niebieskiego (b).

sposoby określania koloru

właściwość color pozwala określić kolor tekstu.
Kolory w CSS można określić na różne sposoby.

	KOLOR CZARNY	KOLOR BIAŁY
nazwa koloru	black	white
wartość rgb	rgb (0, 0, 0)	rgb (255, 255, 255)
kod szesnastkowy	#000000	#ffffff
kod hsl	hsl(0, 0%, 0%)	hsl(0, 100%, 100%)

kolory hsl

w CSS można określać kolory przy użyciu odcienia, nasycenia i jasności

odcień

odcień często jest prezentowany przy użyciu koła, w którym kąt określa kolor. Przyjmuje wartości od 0 do 360.

nasycenie

nasycenie określa ilość szarości w kolorze. Jest wyrażane jako wartość procentowa:
100% - pełne nasycenie koloru
0% - odcień szarości

jasność

jasność określa ilość bieli lub czerni. Jest wyrażana jako wartość procentowa (podobnie do nasycenia):
100% - kolor biały
50% - kolor normalny
0% - kolor czarny

rgba i hsla

kanał alfa

w modelu rgba i hsla ostatnia wartość to kanał alfa – nieprzezroczystość. Przyjmuje wartości z zakresu od 0 do 1.

0.5 – 50% nieprzezroczystości

0.75 – 75% nieprzezroczystości

```
h1 {
        color: rgba(104, 21, 151, 0.781);
        background-color: hsla(120, 50%, 50%, 0.5);
}
```

jednostki

px

należy ich używać wtedy, gdy zależy nam, aby wybrany element zajmował zawsze taki sam obszar na ekranie, niezależnie od jego rozdzielczości i wielkości.

%

rozmiary stylowanego w ten sposób elementu są zależne od wielkości jego rodzica.

vh (viewport height)

wysokość okna przeglądarki. Przeglądarka sprawdza jaką ma aktualnie wysokość i następnie aplikuje ją do stylów CSS.

vw (viewport width)

szerokość okna przeglądarki.

em

jednostka względna.
Rozmiary elementów są zależne od wielkości rodzica

rem

jednostka względna
Rozmiary elementów są zależne od wielkości elementu html.

fonty

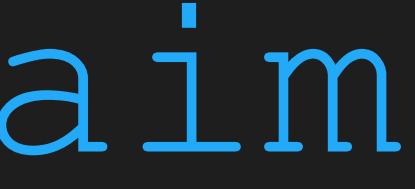
SZERYFOWE

Czcionki szeryfowe charakteryzują się dodatkowymi kreseczkami (szeryfami) na końcach głównych linii tworzących znaki

BEZSZERYFOWE

Czcionki bezszeryfowe mają proste zakończenia linii, dlatego są bardziej przejrzyste

O STAŁEJ SZEROKOŚCI ZNAKÓW



Wszystkie znaki mają taką samą szerokość

stylowanie tekstu

- **font-family**: określenie kroju czcionki
- **font-size**: rozmiar czcionki
- **font-weight**: pogrubienie
- **font-style**: kursywa

stylowanie tekstu

- **text-align**: wyrównanie do prawej, lewej, środka itd.
- **text-decoration**: podkreślenie, przekreślenie itd.
- **text-transform**: zmiana wielkości liter
- **letter-spacing** / **word-spacing**: odstęp pomiędzy literami / wyrazami
- **line-height**: wysokość linii, w której jest tekst
- **direction**: kierunek tekstu
- **text-shadow**: cień
- **text-indent**: wcięcie

google fonts i font awesome

google fonts

największy, darmowy zbiór czcionek w Internecie. Wszystkie czcionki są udostępnione w wersji open-source – mogą być używane bezpłatnie, zarówno w celach komercyjnych, jak i osobistych.

https://fonts.google.com/

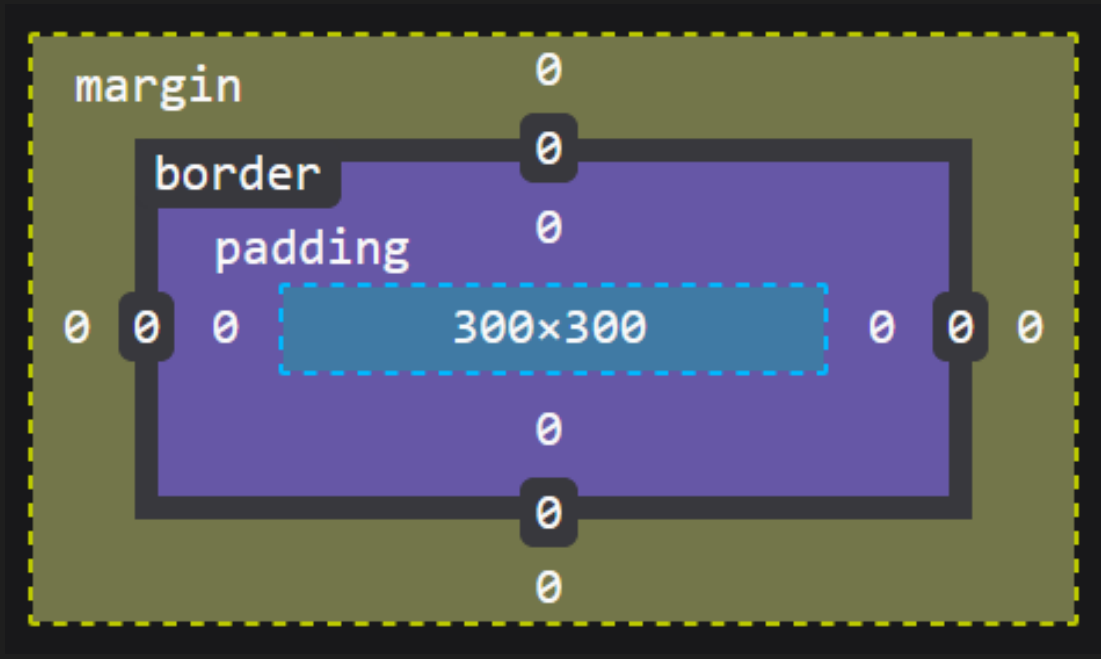
font awesome

font zawierający wektorowe ikony przeznaczone do użytku na stronach internetowych.

https://fontawesome.com/

model pudełkowy

model pudełkowy to schemat, według którego określane są wielkości części składowych elementu. Elementy prezentowane na stronie nie funkcjonują w próżni i każdy z nich generuje prostokątne pole – pudełko (box).



W skład pudełka wchodzą:

- content (zawartość)
- padding (margines wewnętrzny)
- border (obramowanie)
- margin (margines zewnętrzny)

box-sizing

box-sizing: content-box

w skład określonej szerokości wchodzą:

- content

dodanie właściwości padding i border powiększa pudełko.

---

box-sizing: border-box

w skład określonej szerokości wchodzą:

- content
- padding
- border

dodanie właściwości padding i border nie zmienia określonej wielkości elementu.

padding

margines wewnętrzny.

Określa odstęp pomiędzy początkiem elementu, a jego treścią.

pełen zapis deklaracji padding:

```
div {
        padding-top: 10px;
        padding-right: 20px;
        padding-bottom: 10px;
        padding-left: 20px;
}
```

skrócony zapis deklaracji padding:

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

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

border

obramowanie elementu.

Właściwość border składa się z 3 elementów: szerokości obramowania (border-width), stylu obramowania (border-style) oraz koloru (border-color).

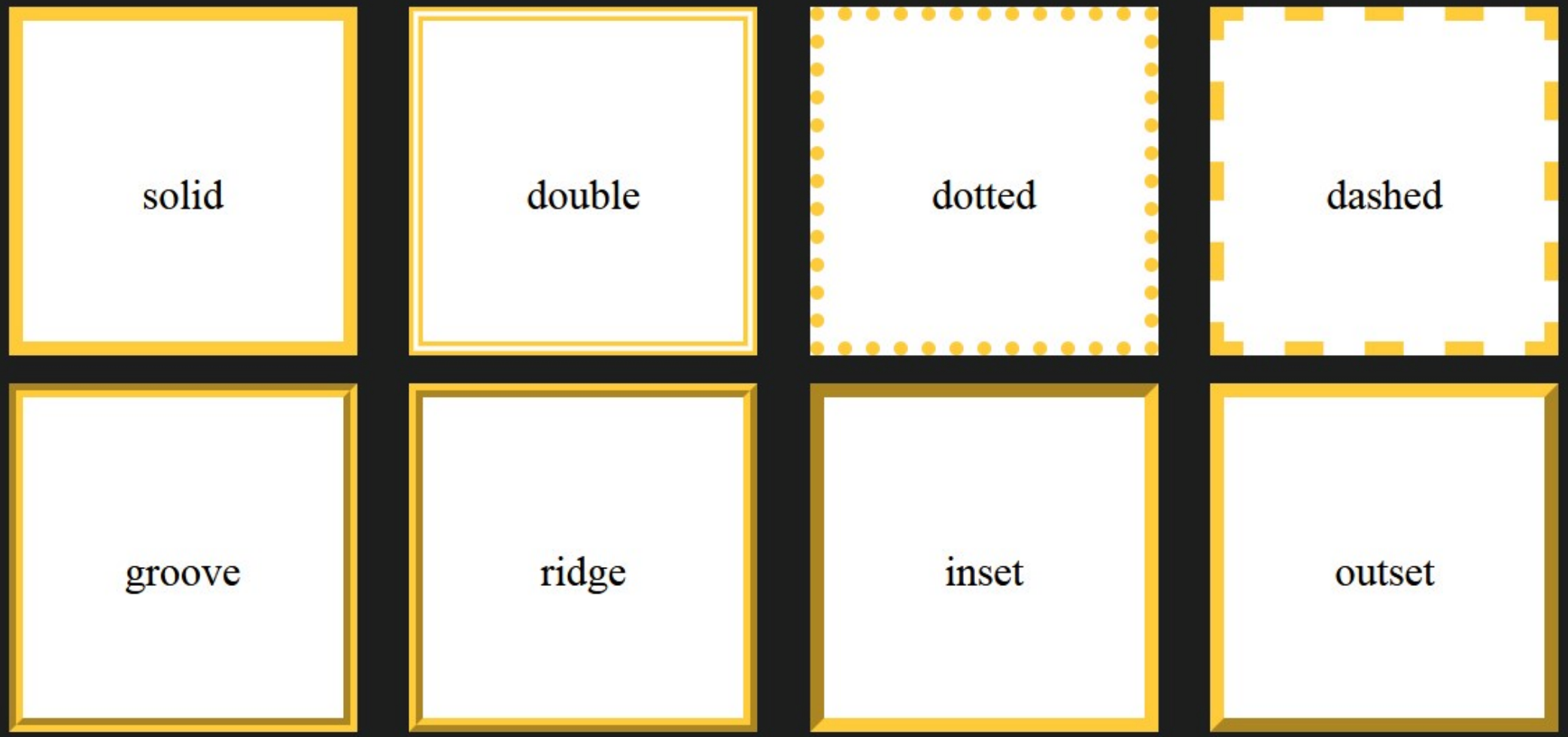
pełen zapis deklaracji border:

```
div {
    border-width: 1px;
    border-style: solid;
    border-color: black;
}
```

skrócony zapis deklaracji border:

```
div {
    border: 1px solid black;
}
```

właściwość border-style i jej wartości



border-radius

zaokrąglanie wierzchołków elementu

pełen zapis deklaracji border-radius:

```
div {
    border-top-right-radius: 10px;
    border-top-left-radius: 10px;
    border-bottom-right-radius: 10px;
    border-bottom-left-radius: 10px;
}
```

skrócony zapis deklaracji border-radius:

```
div {
    border-radius: 10px;
}
```

width, height, border i border-radius

kwadrat

koło

trójkąt równoramienny

trójkąt równoboczny

trójkąt prostokątny równoramienny

trójkąt prostokątny

owal

okrag

margin

margines zewnętrzny.

Określa odstęp pomiędzy elementami

pełen zapis deklaracji margin:

```
div {
        margin-top: 10px;
        margin-right: 20px;
        margin-bottom: 10px;
        margin-left: 20px;
}
```

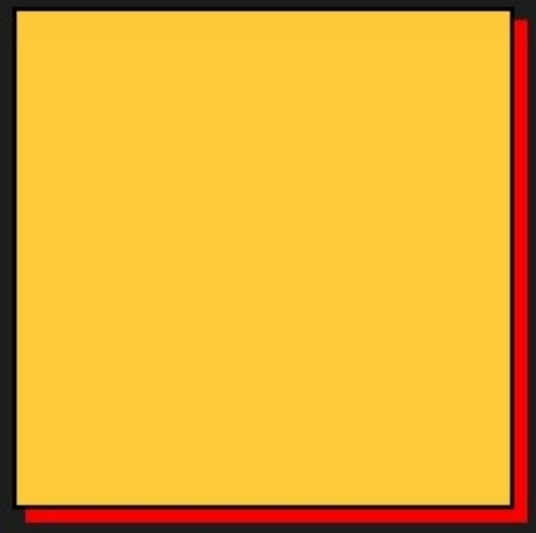
skrócony zapis deklaracji margin:

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

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

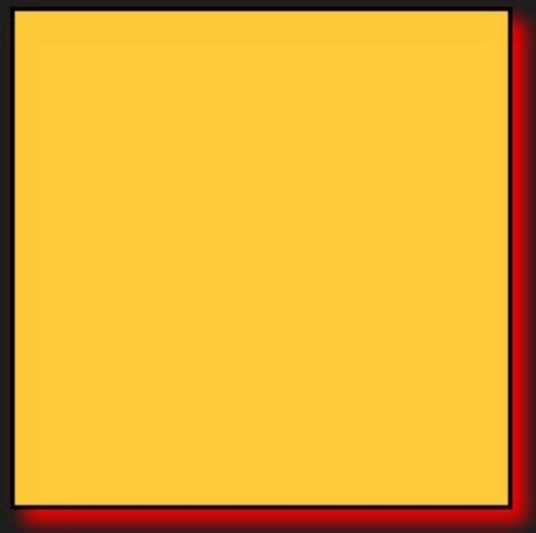
wyśrodkowanie elementów za pomocą marginesów: {margin: 10px auto}

box-shadow

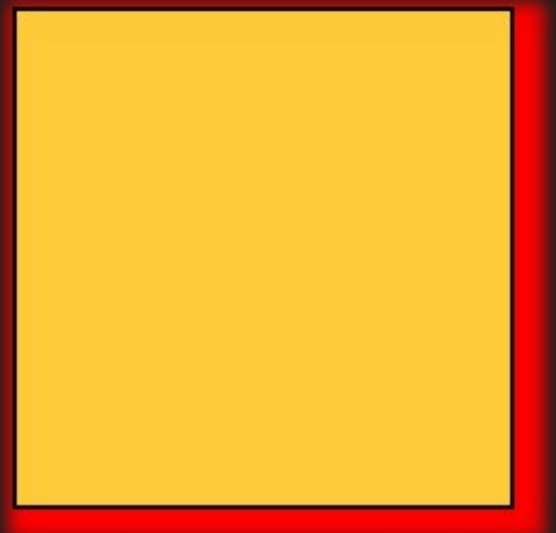
właściwość box-shadow dodaje efekt cienia do elementu



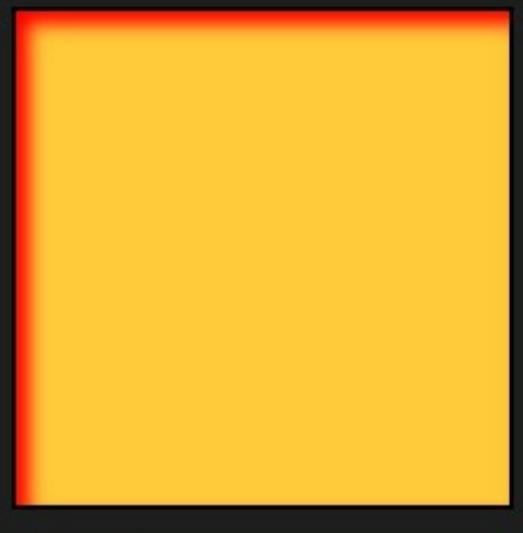
box-shadow: 6px 6px red;



box-shadow: 6px 6px 8px red;



box-shadow: 6px 6px 8px 8px red;



box-shadow: inset 6px 6px 8px red;

display

display określa typ elementów HTML.

Typ elementu ma wpływ między innymi na:

- sposób układania się elementu HTML względem sąsiadujących elementów
- możliwość reagowania elementu HTML na poszczególne właściwości CSS

Najczęściej natrafiane wartości właściwości display:

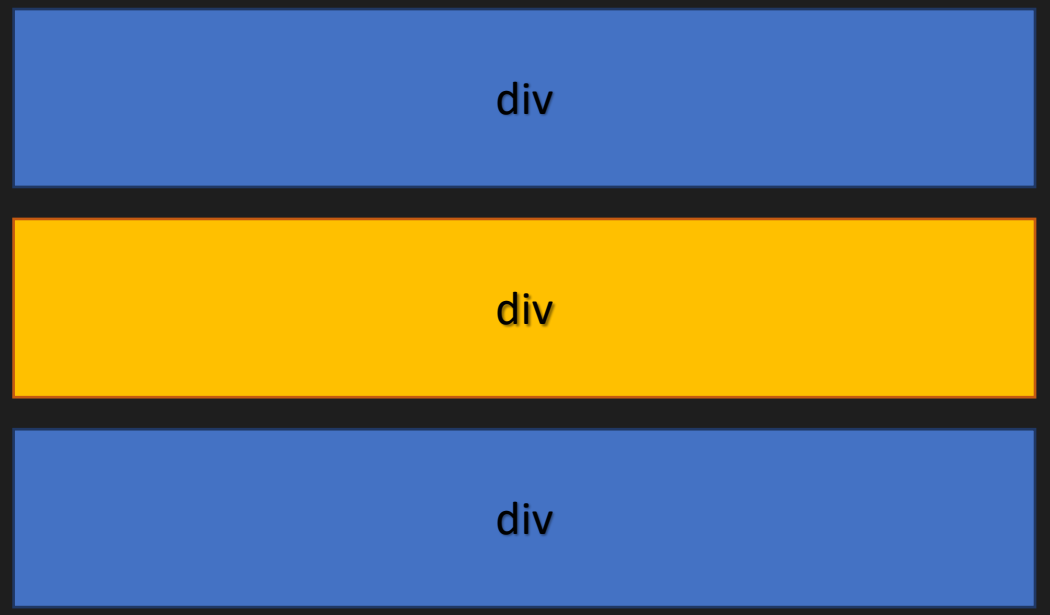
block	inline	inline-block	none
elementy blokowe, można zmieniać ich wielkość	elementy liniowe, nie można zmieniać ich wielkości	elementy zachowują się jak liniowe, ale można manipulować ich wielkością	elementy są niewidoczne

position

position określa, w jaki sposób element jest pozycjonowany w dokumencie. Właściwości top, right, bottom i left określają ostateczne położenie pozycjonowanych elementów.

position: static; - wartość domyślna – elementy są dla siebie widoczne

position: relative;
position: absolute;
position: fixed;
position: sticky;



position: relative;

- element pozostaje na miejscu z punktu widzenia struktury
- można zmieniać jego położenie z punktu widzenia widoku
- elementy pod nim zostają na swoim miejscu


```
div {
    width: 200px;
    height: 200px;
    position: relative;
    top: 100px;
    left: 200px;
    z-index: -1;
}
```

position: absolute;

- element zostaje „wyrwany” z punktu widzenia struktury
- elementy umieszczone pod nim przesuwają się
- można go pozycjonować względem rodzica wykorzystując position: relative na rodzicu
- jeżeli rodzic nie ma właściwości position: relative, to element z właściwością position: absolute jest pozycjonowany względem body

position: fixed;

- elementy z przypisaną właściwością position: fixed są „wyrzucone” z układu i nie wpływają na inne elementy.
- są przyklejone do okna przeglądarki i pozycjonowane względem niej

nawigacja przyklejona 10px od góry strony i 20px od lewej strony

```
div {
    width: 150px;
    height: 50px;
    position: fixed;
    top: 10px;
    left: 20px;
}
```

sprawdzić position: sticky;

z-index

właściwość z-index pozwala określić kolejność elementów na stosie. Im wyższa liczba, tym wyższej w stosie znajduje się element

domyślnie elementy znajdujące się niżej w dokumencie układane są na wierzchu elementów poprzedzających

układ domyślny

kolejność układania z z-index

float

- elementy pływające to elementy dosunięte do lewej lub prawej krawędzi linii
- obok elementów pływających może znajdować się treść

Domyślną wartością właściwości float jest none.

Pozostałe wartości to:

- left – ustawienie do lewej krawędzi
- right – ustawienie do prawej krawędzi
- inherit – przejęcie ustawień float po rodzicach

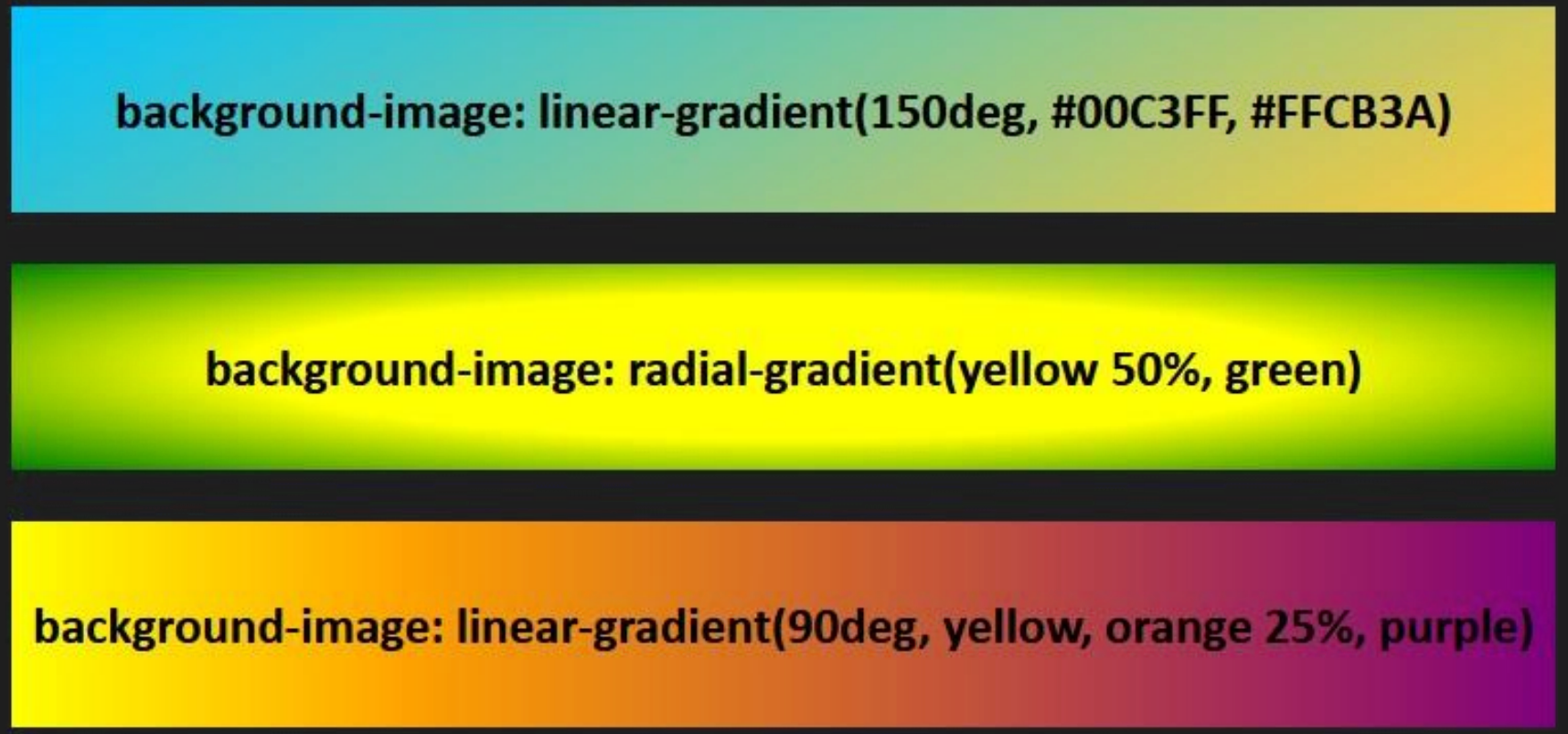
background

właściwość background jest skrótem do ustawienia poszczególnych wartości tła.

Właściwości background i ich wartości domyślne:

- background-image: none
- background-position: 0% 0%
- background-size: auto auto
- background-repeat: repeat
- background-origin: padding-box
- background-clip: border-box
- background-attachment: scroll
- background-color: transparent – kolor tła

gradient kolorów



generator gradientów: https://cssgradient.io/

pseudoelementy i pseudoklasy

pseudoklasy

- pomagają zmienić wygląd elementów HTML np. po najechaniu na nie myszką, po kliknięciu, itp.
- pozwalają zmieniać wygląd określonych elementów
- nie są związane z żadnym konkretnym elementem – mogą dotyczyć każdego

pseudoelementy

- pozwalają dodać wstawkę przed lub za wybranym elementem HTML
- pozwalają ostylować wybrane części elementów HTML np. pierwszą literę

pseudoelementy i pseudoklasy

pseudoklasy

- aby dodać pseudoklasę do selektora należy użyć znaku dwukropka i nazwy pseudoklasy np. a:hover lub dodać ją do utworzonej klasy klasa:first-child

Popularne pseudoklasy:

- :hover – określa hiperłącze, na które został ustawiony kursor myszy
- :visited – określa hiperłącza, które zostały już odwiedzone
- :first-child – określa każdy element HTML będący pierwszym elementem potomnym rodzica
- :last-child –
- :nth-child(n) – określa każdy element HTML, który jest n-tym elementem rodzica

pseudoelementy

- aby dodać pseudoelement do selektora należy użyć podwójnego znaku dwukropka i nazwy pseudoelementu np. h1::after

Popularne pseudoelementy:

- ::before – umieszcza zawartość przed określonym elementem HTML
- ::after – umieszcza zawartość po określonym elemencie HTML
- ::first-letter – określa pierwszą literę wybranego elementu HTML
- ::first-line – określa pierwszą linię wybranego elementu HTML

transition

właściwość transition jest skróconą właściwością dla innych właściwości składowych:

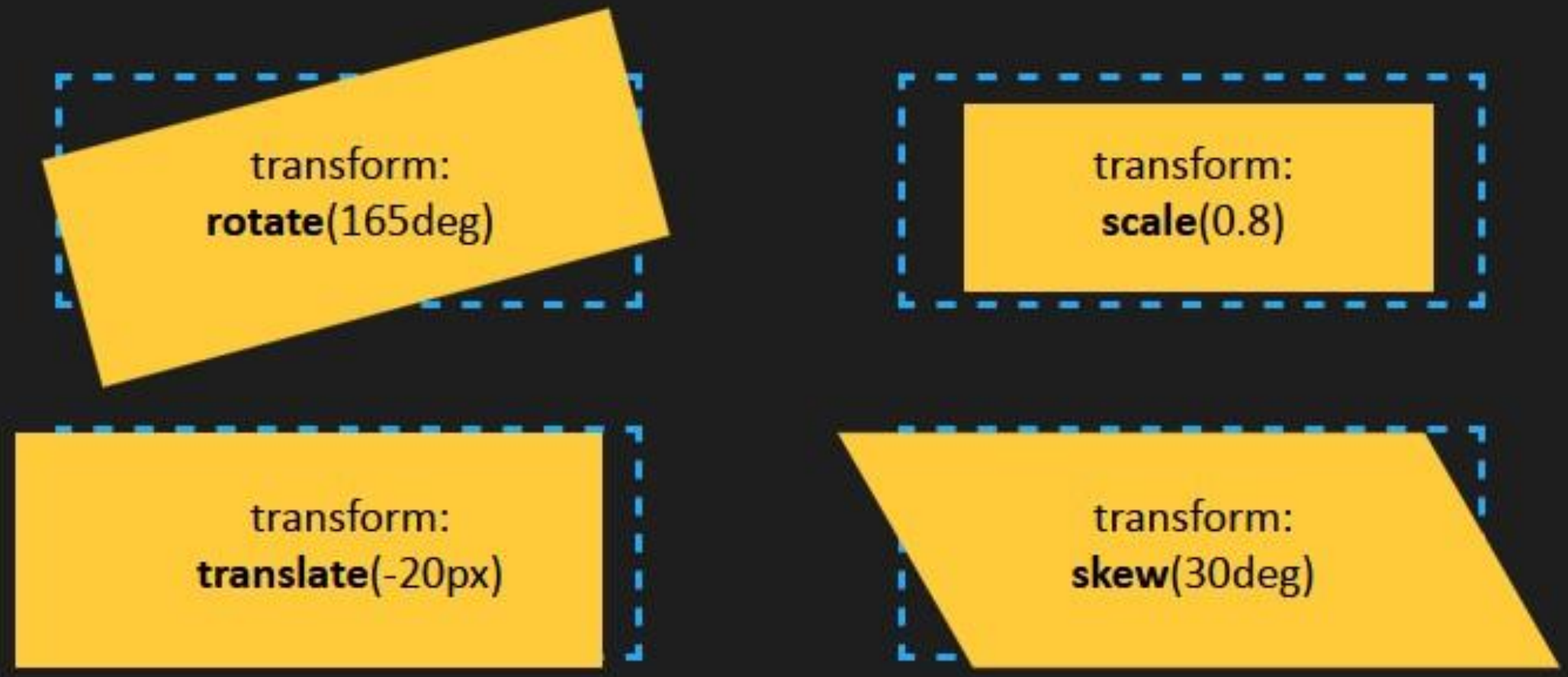
transition-property	określa nazwę właściwości CSS
transition-duration	określa czas trwania efektu przejścia
transition-timing-function	określa krzywą prędkości efektu przejścia
transition- delay	określa opóźnienie rozpoczęcia przejścia

transform

właściwość transform umożliwia transformację elementu, np. obracanie, skalowanie, pochylanie itd.

transform: translate(120px, 50%)	przesunięcie elementu (wzdłuż osi x i osi y)
transform: scale(1.5)	powiększenie elementu
transform: rotate(145deg);	obrót elementu o dany kąt
transform: skew(30deg, 20deg);	pochylenie elementu (wzdłuż osi x i osi y)

transform – przykłady



transform-origin

transform-origin określa punkt względem którego następuje transformacja

wartości domyślne:
transform-origin: 50% 50% - transformacja względem środka elementu

inne przykładowe wartości:
transform-origin: 0 0 – względem początku elementu (lewego, górnego rogu)
transform-origin: 100% 100% - względem prawego dolnego rogu

animacje

```
@keyframes nazwaAnimacji {
    0% {width: 100%;}
    30% {width: 10%;}
    60% {width: 50%;}
    100% {width: 100%;}
}
```

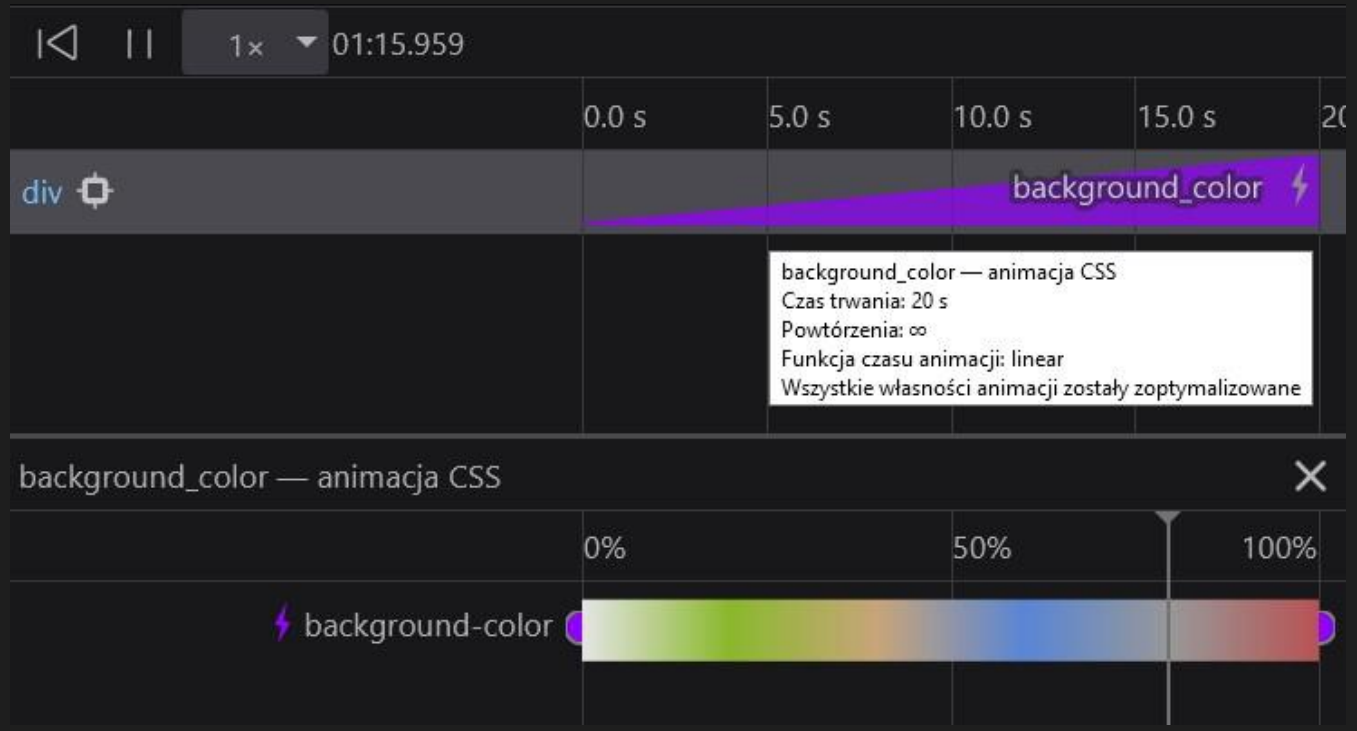
animation

właściwość animation jest skróconą nazwą dla innych właściwości

animation-name	**określa nazwę animacji**
animation-duration	**określa ile sekund lub milisekund trwa animacja**
animation-timing-function	określa krzywą prędkości animacji
animation-delay	określa opóźnienie, zanim rozpocznie się animacja
animation-iteration-count	określa, ile razy animacja powinna być odtwarzana
animation-direction	określa kierunek animacji
animation-fill-mode	określa styl dla elementu, gdy animacja nie jest odtwarzana

analiza animacji

```
div {
  animation: background_color 20s infinite;
}

@keyframes background_color {
  0% {background-color: white;}
  20% {background-color: yellowgreen;}
  40% {background-color: burlywood;}
  60% {background-color: cornflowerblue;}
  80% {background-color: darkgray;}
  100% {background-color: indianred}
}
```



flexbox

flexbox to elastyczny układ pudełkowy

- kluczową cechą flexbox jest to, że elementy w układzie flex mogą rosnąć i kurczyć się. Przestrzeń można przypisać do samych elementów lub rozdzielić między nimi lub wokół nich.

- flexbox umożliwia również wyrównanie elementów na osi głównej lub poprzecznej, zapewniając tym samym wysoki poziom kontroli nad rozmiarem i wyrównaniem grupy elementów.

kontener i elementy elastyczne

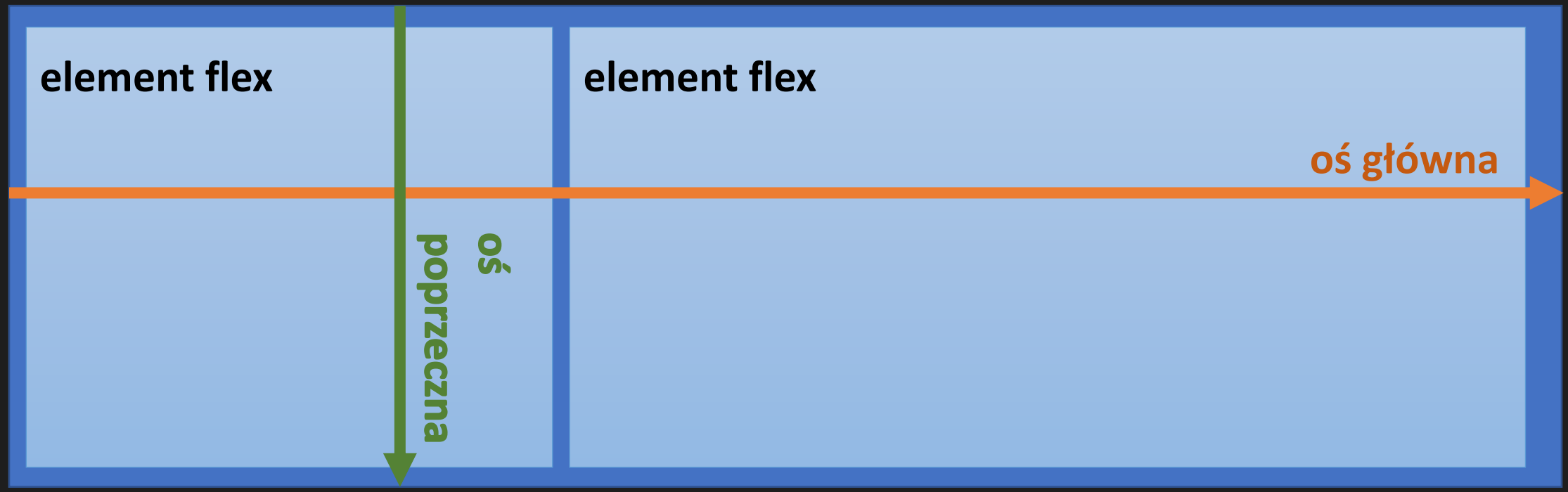
po nadaniu elementowi section właściwości display: flex staje się on kontenerem, a jego dzieci elementami elastycznymi

```
<section>
        <div>1</div>
        <div>2</div>
        <div>3</div>
</section>
```

właściwości rodziców

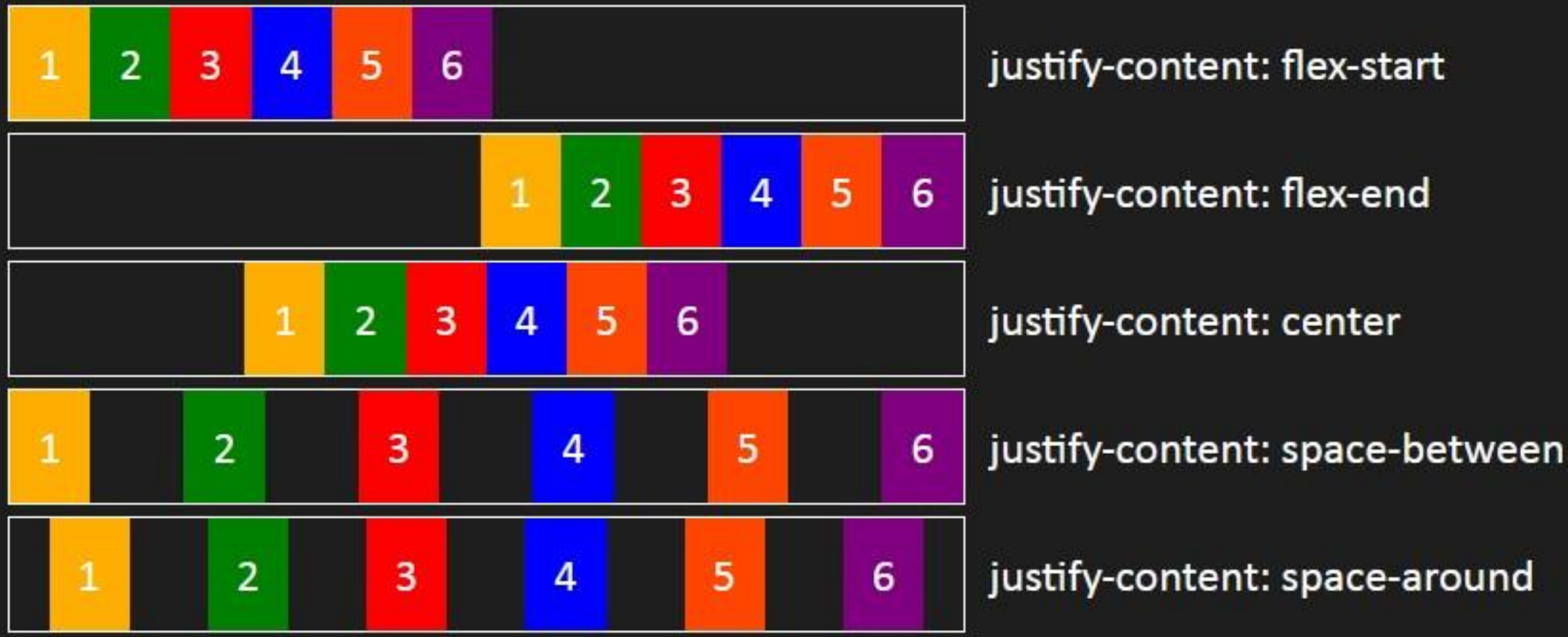
- display: flex (niezbędne, żeby flex zaczął działać)
- flex-direction: row
- flex-wrap: nowrap
- align-items: stretch
- justify-content: flex-start
- align-content: stretch

flex-direction: row;



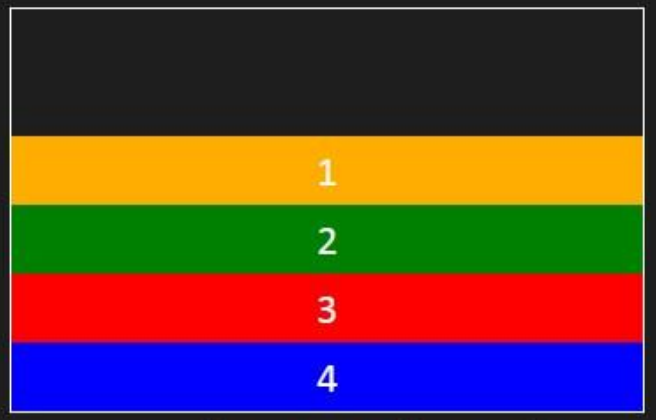
flex-direction: column;

justify-content w układzie row

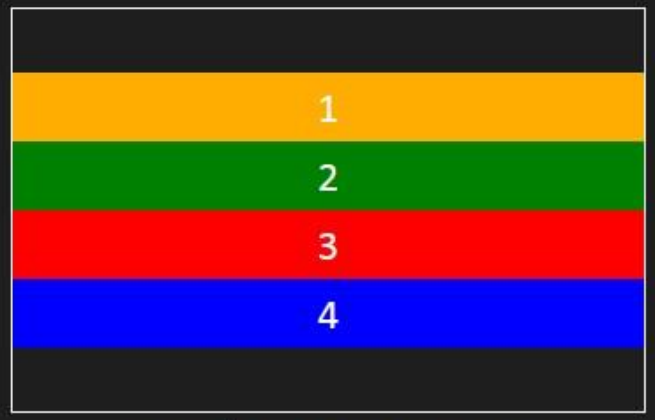


justify-content w układzie column

justify-content: flex-start



justify-content: flex-end



justify-content: center

justify-content: space-between

justify-content: space-around

właściwości elementów elastycznych

jeśli kontener ma display: flex, to możemy dla jego dzieci (elementów elastycznych) określić specjalne właściwości:

- flex-grow : 0
- flex-shrink: 1
- flex-basis: auto
- align-self: auto
- order: 0

flex-grow

za pomocą właściwości flex-grow oraz dowolnej liczby nieujemnej można określić proporcję rozrostu elementów elastycznych

BEM

BEM – to metodologia, która pomaga tworzyć komponenty wielokrotnego użytku przez nadawanie im klas

Reguły BEM

- nazwa klasy informuje za co odpowiada dana część interfejsu
- nazwę klasy, która składa się z dwóch lub więcej wyrazów rozdziela się myślnikiem lub notacją wielbłądzią
- kod CSS buduje się tylko przez nazwy klas (bez zagnieżdżeń znaczników HTML)

http://getbem.com/

BEM

B

block (komponent) – niezależny element interfejsu

```
<nav class="main-menu">…</nav>
```

E

element (część komponentu) – zależny od bloku, nie występuje poza nim

```
<nav class="main-menu">
    <ul class="main-menu__list">…</ul>
</nav>
```

M

modifier – niestandardowy blok lub element (zmieniony), nie występuje samodzielnie

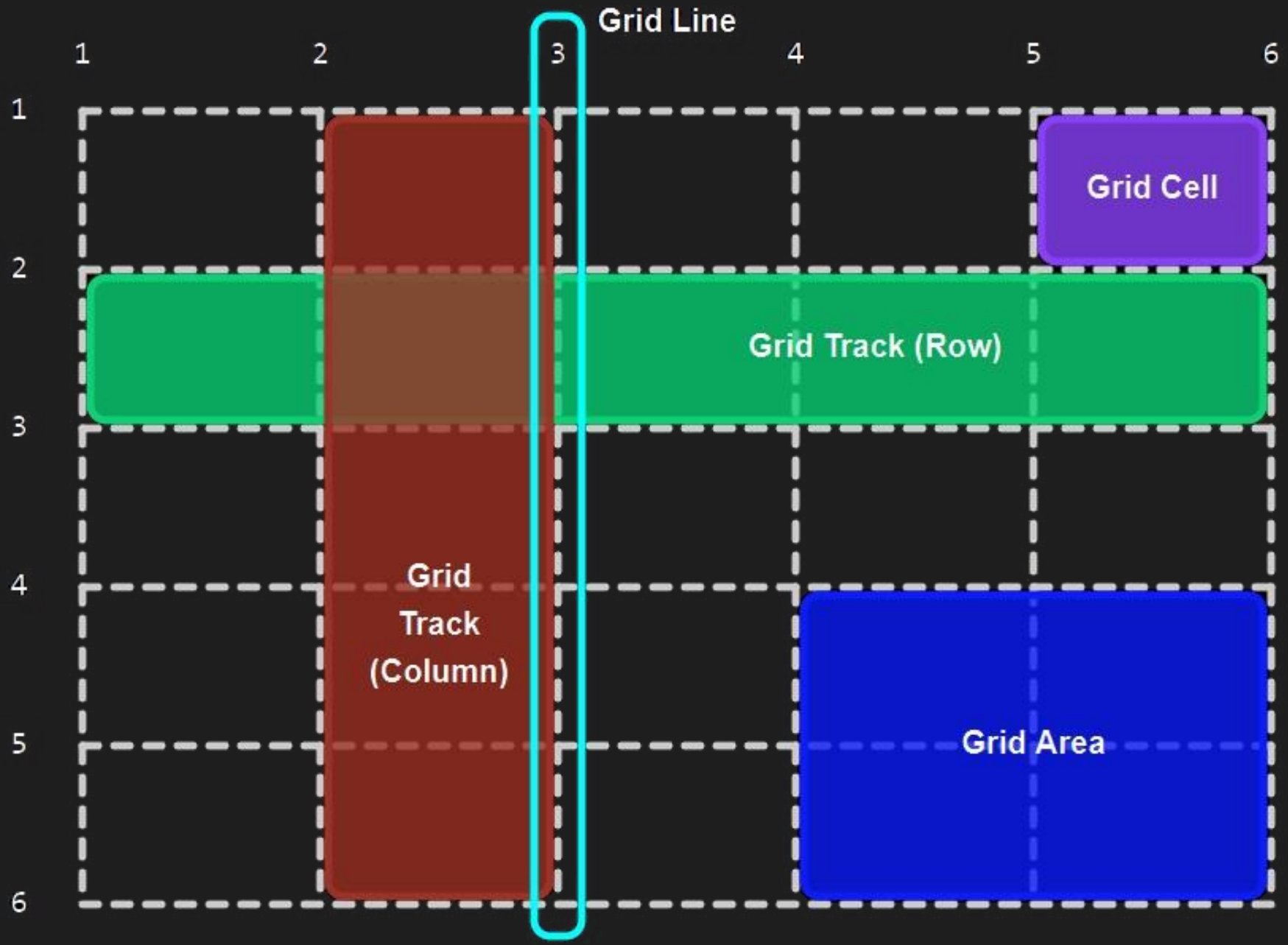
```
<nav class="main-menu">
 <li class="main-menu__item"></li>
 <li class="main-menu__item  main-menu__item--visited"></li>
</nav>
```

grid

grid – system do budowania layoutów

Podstawowe pojęcia w systemie grid:

- grid line
- grid cell
- grid area
- grid track (row)
- grid track (column)
- grid gutter


Grid Line
1
2
3
4
5
6
Grid Cell
Grid Track (Row)
Grid
Track
(Column)
Grid Area

grid

- display: grid
- grid-template-columns: podział na kolumny
  jednostka fr - frakcje
- grid-template-rows: podział na wiersze
- grid-column-gap: przerwa między kolumnami (gutter)
- grid-row-gap: przerwa między wierszami

- zapis skrócony – grid-gap: 20px 10px;

grid – właściwości dzieci

- grid-row-start: 1
- grid-row-end: 2
- zapis skrócony – grid-row: 1/2

- grid-column-start: 1
- grid-column-end: 2
- zapis skrócony – grid-column: 1/2

RWD

RWD (Responsive Web Design) – technika projektowania stron internetowych w taki sposób, że mogą one dostosowywać się do szerokości ekranów, na których są wyświetlane np. laptopów, smartfonów czy tabletów.

JAN 2020

SHARE OF WEB TRAFFIC BY DEVICE

EACH DEVICE'S SHARE OF TOTAL WEB PAGES SERVED TO WEB BROWSERS IN DECEMBER 2019

MOBILE PHONES	LAPTOPS & DESKTOPS	TABLET COMPUTERS	OTHER DEVICES
			
53.3%	**44.0%**	**2.7%**	**0.07%**
DEC 2019 vs. DEC 2018:	DEC 2019 vs. DEC 2018:	DEC 2019 vs. DEC 2018:	DEC 2019 vs. DEC 2018:
+8.6%	**-6.8%**	**-27%**	**-30%**

SOURCE: STATCOUNTER (ACCESSED JANUARY 2020). FIGURES REPRESENT EACH DEVICE'S SHARE OF WEB PAGES SERVED TO WEB BROWSERS ONLY. **NOTE:** FIGURES FOR DEVICE SHARE ARE FOR DECEMBER 2019; ANNUAL CHANGE FIGURES COMPARE MONTHLY SHARE VALUES FOR DECEMBER 2019 VS. DECEMBER 2018.

projektowanie i tworzenie stron WWW

RWD vs. AWD

RWD - Responsive Web Design

podstawowym założeniem **responsywnego** projektu jest napisanie takiego arkusza stylów by elementy były elastyczne i dostosowywały się do dostępnej szerokości, a w regułach *@media* znalazło się jak najmniej deklaracji.

AWD - Adaptive Web Design

stosując podejście **adaptywne** tworzymy tak naprawdę kilka layoutów i stosując **media queries** zmieniamy wielkości elementów oraz układ strony, gdy okno przeglądarki osiągnie określony breakpoint


RESPONSIVE
Logo
RESPONSIVE
RESPONSIVE
RESPONSIVE


ADAPTIVE
ADAPTIVE
ADAPTIVE
Logo
ADAPTIVE
ADAPTIVE
ADAPTIVE
ADAPTIVE

media queries

zapytania o media są podstawą technologią responsywnego projektowania witryn internetowych.

Dwa sposoby użycia:

- media query użyta w elemencie link:

```
<link rel="stylesheet" media="screen and (min-width:576px)" href="./CSS/small.css">
```

- media query w środku arkusza stylów:

```
@media screen and (min-width: 576px) {
        p {
                font-size: 15px;
        }
}
```

Responsive Web Design

mobile first i desktop first

mobile first

technika, według której pierwszym krokiem jest budowanie widoku mobilnego

wykorzystując media queries należy określić minimalną szerokość i stopniowo przechodzić do większych rozmiarów

```
@media screen and (min-width: 576px) {
        p {
                font-size: 15px;
        }
}
```

desktop first

technika, według której pierwszym krokiem jest budowanie widoku desktopowego

wykorzystując media queries należy określić maksymalną szerokość i stopniowo przechodzić do mniejszych rozmiarów

```
@media screen and (max-width: 768px) {
        p {
                font-size: 15px;
        }
}
```

zalety podejścia mobile first

Dlaczego Mobile First Design jest tak ważny?

- uzyskujemy bardziej dopracowaną wersję mobilną, z której korzysta większy procent użytkowników
- wciąż kreujemy w pełni funkcjonalną wersję desktop, z adekwatnymi skryptami i układami.
- uzyskujemy szybsze wczytywanie się stron na telefonach (plik CSS posiada najmniejszą wagę dla telefonów, a największą dla desktopów)
- bazowanie na prostym layoucie oraz utrzymywanie mniejszej ilości bardziej przejrzystego kodu pozwala w szybszym tempie rozwijać layout dla urządzeń o większych ekranach

dobre praktyki pisania kodu

Jak porządkować kod CSS

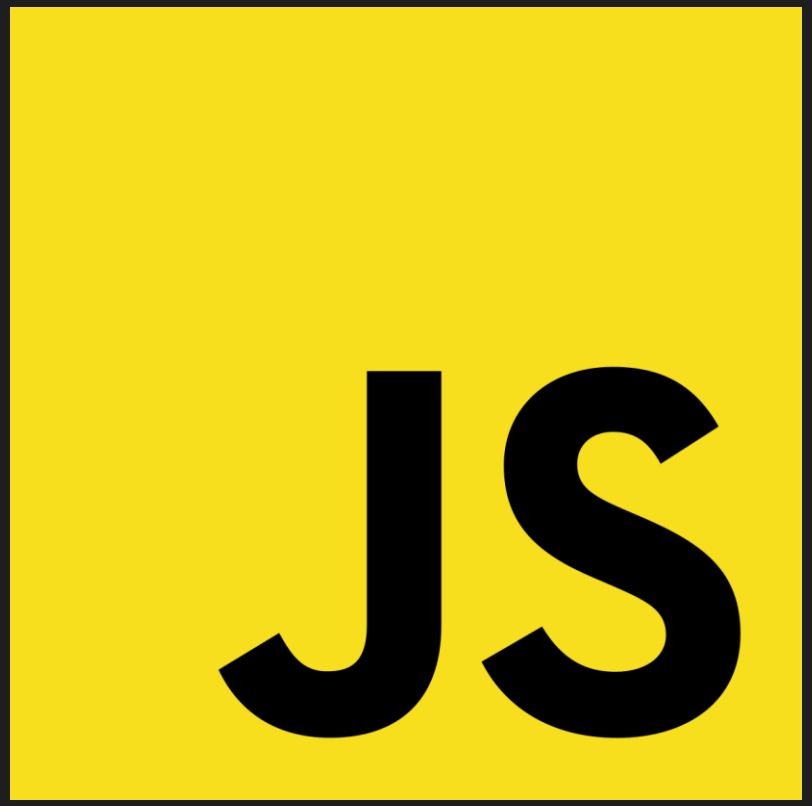
```
div {
        position; top/bottom; left/right

        display; box-sizing; padding/border/margin; width; height

        color; background

        font-size; font-weight i inne właściwości fontów

        cursor; transition
}
```

ciekawe linki

- https://developers.google.com/speed/pagespeed/insights/?hl=pl
- https://www.quackit.com/css/color/charts/css_color_names_chart.cfm
- https://css-tricks.com/almanac/properties/c/cursor/
- http://getbem.com/

JavaScript

wprowadzenie



- **skryptowy** język programowania, który umożliwia obsługę dynamicznego tworzenia treści na stronie internetowej, kontrolowanie multimediów, animację obrazów itp.
- JavaScript jest językiem **interpretowanym** (przeciwieństwo to język kompilowany) - kod jest wykonywany od góry do dołu i wynik jest zwracany natychmiastowo
- standardem dla JavaScript jest ECMAScript. W 2015 ECMA International opublikował szóstą główną wersję ECMAScript, która oficjalnie nazywa się ECMAScript 2015, ale początkowo była nazywana też ECMAScript 6 lub ES6

umieszczanie kodu JavaScript w HTML

I sposób – skrypty osadzone

```
<body>
   <!-- kod HTML -->
   <script>
      //kod JavaScript
   </script>
</body>
```

II sposób – zewnętrzny skrypt

```
<body>
   <!-- kod HTML -->
   <script src="script.js"></script>
</body>
```

zmienne

zmienna przechowuje dane w pamięci skryptu tak, aby można było te dane wykorzystać.

zmienne

zmienna stała (ES6)

const

- może być zadeklarowana i zainicjalizowana tylko raz
- nie można przypisać do niej nowej wartości

zmienna (ES6)

let

- może być nadpisana
- może być zadeklarowana tylko raz
- nie musi być inicjalizowana podczas deklaracji

zmienna

var

- zapis sprzed ES6
- podlega hoistingowi

typy danych

proste

- string
- number
- boolean
- undefined
- null
- symbol (ES6)

złożone

- obiekt
- funkcja
- tablica

typeof – operator, który zwraca informację o typie wartości zmiennej

operatory

- przypisania
- arytmetyczne
- porównania
- logiczne

operatory przypisania

```
let x = 5; // operator przypisania
let y = 5;

x += y; //x = x + y
console.log(x);

x -= y; //x = x - y

x *= y; //x = x * y

x /= y //x = x / y

x %= y; //x = x % y
```

operatory arytmetyczne

```
let x = 10;
let y = 15;

const add = x + y; //operator dodawania
console.log(add);

const sub = x - y;  //operator odejmowania
console.log(sub);

const multiply = x * y;  //operator mnożenia
console.log(multiply);

const divide = x / y;  //operator dzielenia
console.log(divide);
```

operatory arytmetyczne

```
let x = 10;
let y = 15;

const modulo = y % x; //reszta z dzielenia
console.log(`modulo: ${modulo}`);

x++;  //inkrementacja
console.log(x);

x--;  //dekrementacja
console.log(x);
```

operatory porównania

==

(równy z)

porównuje dwie wartości (liczby, stringi, wartości boolean) i sprawdza czy są takie same (konwersja typów)

wynikiem porównania ”3” == 3 jest true

wynikiem porównania 0 == true jest false

!=

(różny)

porównuje dwie wartości (liczby, stringi, wartości boolean) i sprawdza czy są różne (konwersja typów)

wynikiem porównania 0 != true jest true

wynikiem porównania ”3” != 3 jest false

>

(większy niż)

sprawdza czy liczba po lewej stronie jest ***większa niż*** liczba po prawej stronie

wynikiem porównania 4 > 3 jest true

wynikiem porównania 3 > 4 jest false

<

(mniejszy niż)

sprawdza czy liczba po lewej stronie jest ***mniejsza niż*** liczba po prawej stronie

wynikiem porównania 2 < 3 jest true

wynikiem porównania 5 < 4 jest false

===

(identyczny z)

porównuje dwie wartości i sprawdza czy zarówno wartość, jak i typ danych są takie same

wynikiem porównania 3 === 3 jest true

wynikiem porównania ”3” === 3 jest false

!==

(nieidentyczny z)

porównuje dwie wartości i sprawdza czy zarówno wartość, jak i typ danych ***nie*** są takie same

wynikiem porównania ”3” !== 3 jest true

wynikiem porównania 3 !== 3 jest false

>=

(większy niż lub równy)

sprawdza czy liczba po lewej stronie jest ***większa niż lub równa*** liczbie po prawej stronie

wynikiem porównania 3 >= 3 jest true

wynikiem porównania 3 >= 4 jest false

<=

(mniejszy niż lub równy)

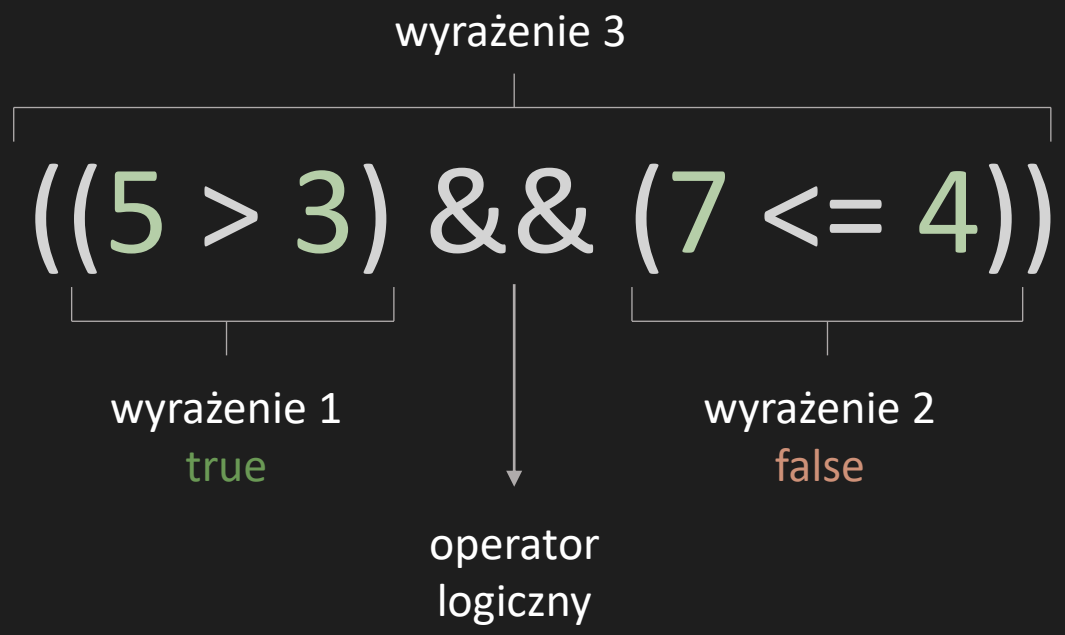
sprawdza czy liczba po lewej stronie jest ***mniejsza niż lub równa*** liczbie po prawej stronie

wynikiem porównania 2 <= 3 jest true

wynikiem porównania 5 <= 4 jest false

operatory logiczne

operatory logiczne pozwalają na porównywanie wyników działania co najmniej dwóch operatorów porównania



operatory logiczne

logiczny iloczyn (and)

&&

(5 > 3) && (7 <= 4)

true && true zwraca true

true && false zwraca false

false && true zwraca false

false && false zwraca false

logiczna suma (or)

||

(5 > 3) || (7 <= 4)

true || true zwraca true

true || false zwraca true

false || true zwraca true

false || false zwraca false

logiczna negacja (not)

!

!(2 < 1)

!true zwraca false

!false zwraca true

instrukcje warunkowe

instrukcje warunkowe pozwalają na podejmowanie w kodzie decyzji dotyczących dalszego sposobu jego wykonania

```
if (warunek) {
        blok kodu do wykonania, jeżeli wartością warunku jest TRUE
} else {
        blok kodu do wykonania, jeżeli wartością warunku jest FALSE
}
```

instrukcja warunkowa if

instrukcja if

instrukcja if dokonuje oceny warunku (sprawdza go). Jeżeli wartością warunku będzie true, to nastąpi wykonanie poleceń znajdujących się w bloku.

```
const password = 'abcd12345@#';

if (password.length > 10) {
    console.log('Twoje hasło jest dobre');
};
```

konstrukcja if else

konstrukcja if else sprawdza warunek. Jeżeli wartością warunku będzie true, to nastąpi wykonanie pierwszego bloku kodu. Jeśli wynikiem jest false, wykonany będzie drugi blok kodu.

```
if (password.length > 15) {
    console.log('Twoje hasło jest dobre');
} else {
    console.log('Twoje hasło jest za krótkie');
};
```

instrukcja warunkowa if

klauzula else if

```
const password = 'abcd12345@#';
const passLength = 15;

if (password.length > passLength && password.includes('#')) {
    console.log('Twoje hasło jest bardzo dobre');
} else if (password.length > passLength) {
    console.log('Twoje hasło jest dobre');
} else if (password.length > 2 && password.length < passLength) {
    console.log('Twoje hasło jest średnie');
} else {
    console.log('Twoje hasło jest za krótkie');
};
```

zagnieżdżanie instrukcji warunkowych

zagnieżdżenie if

```
const name = 'Ola';
const age = 30;

if (name) {
    if (age >= 18) {
        console.log(`witaj ${name}, możesz wejść`);
    } else {
        console.log(`witaj ${name}, masz za mało lat`);
    }
} else {
    console.log('nie wiem kim jesteś');
};
```

klauzula else if

```
const name = 'Ola';
const age = 30;

if (name && age >= 18) {
    console.log(`witaj ${name}, możesz wejść`);
} else if (name && age < 18) {
    console.log(`witaj ${name}, masz za mało lat`);
} else {
    console.log('nie wiem kim jesteś');
};
```

instrukcja warunkowa switch

konstrukcja switch rozpoczyna się od zmiennej nazywanej wartością switch. Każdy blok case zawiera możliwą wartość dla tej zmiennej i uruchamia zdefiniowane w nim polecenia, jeśli nastąpi dopasowanie wartości zmiennej.

```
const day = 'czwartek';
```

wartość switch, która będzie porównywana

```
switch (day) {
    case 'czwartek':
        console.log('dzisiaj jest czwartek');
        break;

    default:
        console.log('jest weekend');
};
```

wartość do porównania z argumentem instrukcji

kod, który ma się wykonać

zatrzymanie działania instrukcji

operator warunkowy (ES6)

operator warunkowy to skrócona wersja warunku **if**, który zwraca wartość, którą można przypisać

```
const x = 22;

const isEven = (x % 2 === 0) ? "parzysta" : "nieparzysta";

console.log(isEven);
```

warunek

wykonanie jeżeli wartością warunku jest true

wykonanie jeżeli wartością warunku jest false

instrukcja if vs. operator warunkowy

instrukcja warunkowa if

```
const y = 25;
let text;

if (y < 50) {
    text = `${y} jest mniejszy niż 50`;
} else {
    text = `${y} jest większy niż 50`;
};
```

operator warunkowy

```
const y = 25;
let txt;

txt = (y > 50) ? `${y} jest większy niż 50` :
`${y} jest mniejszy niż 50`;
```

pętle

słowo kluczowe

warunek (licznik)

```
for (let i = 0; i <= 10; i++) {
    console.log(i);
}
```

kod do wykonania w trakcie pętli

licznik pętli

Pętla for używa licznika jako warunku, który składa się z trzech poleceń

INICJALIZACJA

utworzenie zmiennej i przypisanie jej wartości 0. Ta zmienna często ma nazwę *i* oraz działa w charakterze licznika.

let i = 0

zmienna jest tworzona tylko w trakcie pierwszej iteracji pętli

WARUNEK

pętla kontynuuje działanie aż do chwili, gdy licznik osiągnie określoną wartość

i < 10

zmienna i miała początkowo wartość 0. W omawianym przypadku pętla zostanie wykonana 10 razy.
Warunek może przechowywać także zmienną przechowującą liczbę.

UAKTUALNIENIE

w trakcie każdej iteracji pętli następuje dodanie wartości 1 do licznika

i++

wartość licznika jest zwiększona za pomocą operatora inkrementacji (i++)

fizzbuzz

pierwotnie test *fizzbuzz* zyskał popularność jako gra przeznaczona dla dzieci, mająca na celu nauczenie ich dzielenia. Dzieci wymieniały kolejne liczby całkowite. Cała zabawa polegała na tym, że jeżeli liczba była podzielna przez 3, to – zamiast podania samej liczby – trzeba było krzyknąć fizz, jeżeli była podzielna przez pięć, to buzz, a jeżeli przez trzy i pięć, to fizzbuzz.

wersja dla programistów:

wypisz wszystkie liczby od 1 do 50, jednak jeżeli liczba jest podzielna przez:

- 3 – wypisz *„fizz”*,
- 5 – wypisz *„buzz”*,
- 3 i 5 wypisz *„fizzbuzz”*.

fizzbuzz - rozwiązanie

```
for (let i = 1; i <= 50; i++) {

   let output;

   if ((i % 3 === 0) && (i % 5 === 0)) {
      output = 'fizzbuzz';
   } else if (i % 3 === 0) {
      output = 'fizz';
   } else if (i % 5 === 0) {
      output = 'buzz';
   } else {
      output = i;
   };
   console.log(output);
}
```

tablice

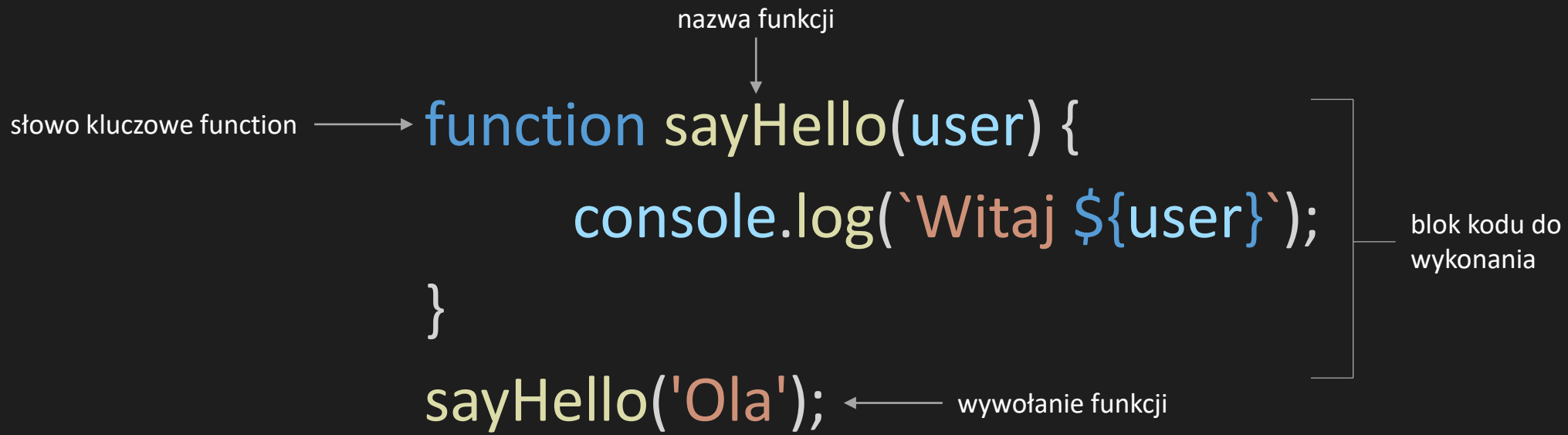
- tablice są uporządkowanymi zbiorami zawierającymi dane
- każda wartość z tablicy ma swój indeks (liczony od 0)

tworzenie tablicy

```
const color = ['blue', 'green', 'gold'];
```

funkcje

funkcja pozwala na zgrupowanie serii poleceń odpowiedzialnych za wykonanie określonego zadania.



zakres zmiennej

Zakres globalny

zakres globalny posiada zmienna, której można używać w dowolnym miejscu w skrypcie

Zakres lokalny

zakres lokalny posiada zmienna, która jest dostępna jedynie w ramach jednej funkcji

zmiana zakresu z lokalnego na globalny

jeżeli zmienna zadeklarowana wewnątrz funkcji ma być widoczna globalnie, to podczas jej deklarowania należy pominąć słowo kluczowe

```
function funkcja() {
    liczba1 = 10
}

funkcja()
console.log(liczba1);
```

zdarzenia

zdarzenie jest akcją, którą można wykryć za pomocą JavaScriptu

zdarzenie	opis zdarzenia
onclick	Kliknięto dany obiekt
onkeydown	Klawisz na klawiaturze został wciśnięty
onkeyup	Klawisz na klawiaturze został puszczony
onload	Zakończyło się ładowanie strony lub obrazka
onmousedown	Naciśnięto przycisk myszy
onmousemove	Wskaźnik myszy poruszył się
onmouseover	Wskaźnik myszy poruszył się nad elementem
onmouseup	Puszczono przycisk myszy
onsubmit	Kliknięto przycisk wysyłania formularza

zdarzenia

istnieją 3 sposoby dodawania metod obsługi zdarzeń do elementów znajdujących się na stronie:

- zapisanie ich jako atrybuty HTML
- zapisanie ich jako metody przyłączonej do elementu
- wykorzystanie funkcji addEventListener

zdarzenia jako atrybuty HTML

funkcję uruchamianą w celu obsługi zdarzenia można zapisać w atrybucie elementu HTML

```
<button onclick="myFunction()">witaj na stronie</button>
```

element HTML

zdarzenie

funkcja, która zostanie wywołana po nastąpieniu zdarzenia

zdarzenia jako metoda

```
window.onclick = myFunction;
```

element — zdarzenie — funkcja, która zostanie wywołana po nastąpieniu zdarzenia

to rozwiązanie ma jedną istotną wadę: pozwala na przyłączenie tylko jednej funkcji do danego zdarzenia

addEventListener

metoda addEventListener pozwala zarejestrować obserwatora zdarzeń dla określonego typu zdarzenia na wybranym elemencie

```
element.addEventListener('click', textIncrease);
```

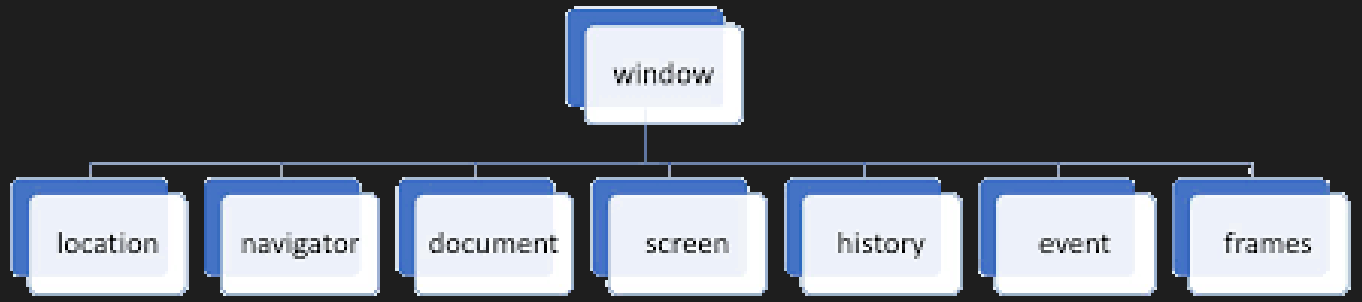
element, na który ustawiana jest metoda addEventListener

zdarzenie

funkcja, która zostanie wywołana po nastąpieniu zdarzenia

obiektowy model przeglądarki

obiektowy model przeglądarki internetowej tworzy model dla okna lub karty przeglądarki internetowej



na samej górze modelu znajduje się obiekt window przedstawiający bieżące okno lub kartę przeglądarki internetowej. Jego obiekty potomne przedstawiają pozostałe funkcje przeglądarki

DOM

DOM – Document Object Model – to reprezentacja dokumentu HTML w przeglądarce

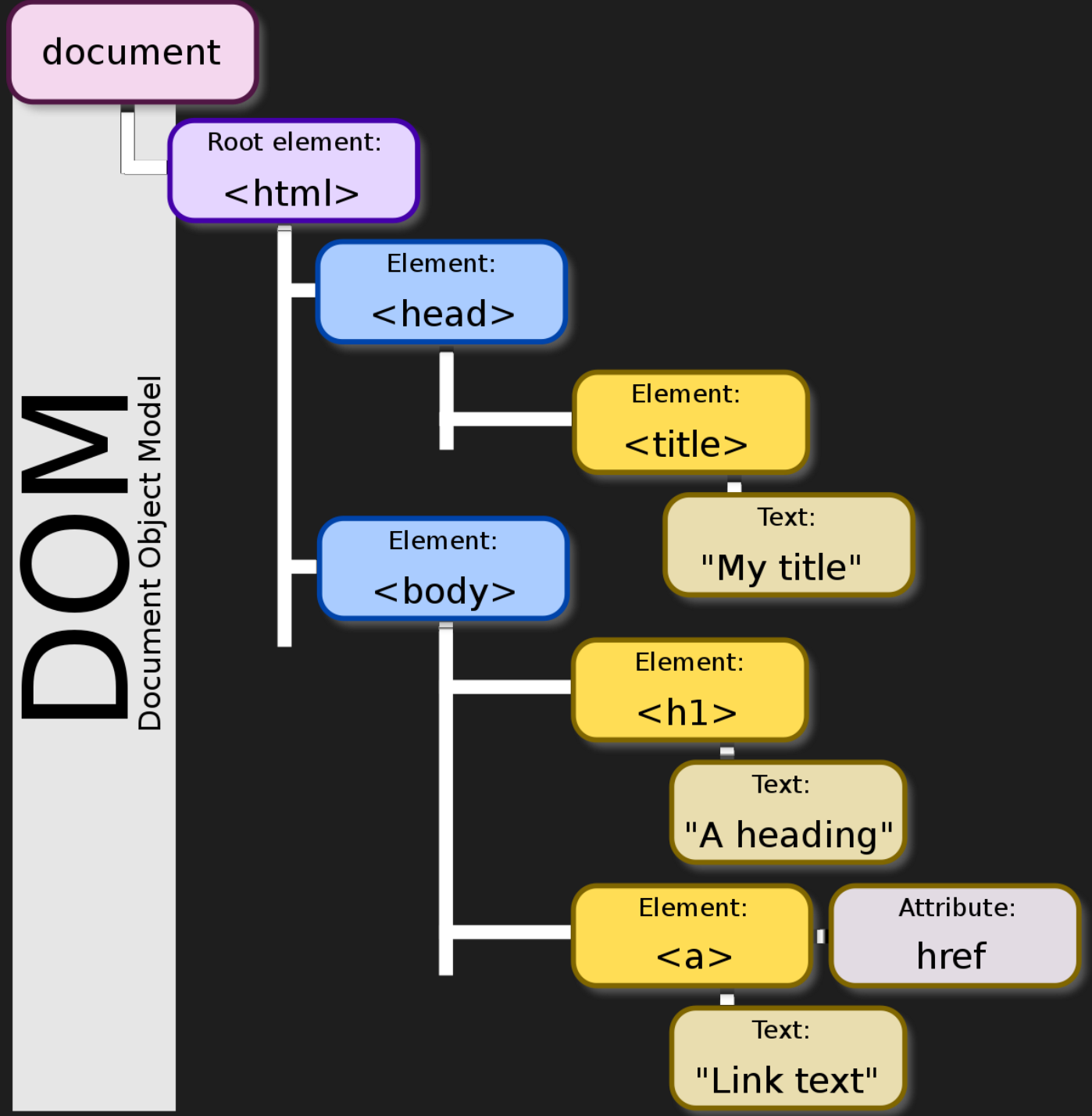
- elementy strony są węzłami, które mogą być pobierane i modyfikowane

Metody pobierające jeden element:

```
document.querySelector('.klasa');
document.getElementById('id_nazwa');
```

Metody pobierające wiele elementów

```
document.querySelectorAll();
document.getElementsByClassName();
document.getElementsByTagName();
```


DOM
Document Object Model
document
Root element:
<html>
Element:
<head>
Element:
<title>
Text:
"My title"
Element:
<body>
Element:
<h1>
Text:
"A heading"
Element:
<a>
Attribute:
href
Text:
"Link text"

Canvas

czym jest canvas

<canvas> jest elementem HTML, który tworzy powierzchnię rysunkową i może być używany do tworzenia grafik, wykresów, prostych i zaawansowanych animacji przy użyciu skryptów (często JavaScript).

domyślny rozmiar kanwy to 300px × 150px

podstawowe użycie

element canvas posiada dwa atrybuty określające szerokość i wysokość płótna.

można go stylizować za pomocą CSS jak każdy obraz, określając właściwości takie jak: margin, border, background itp.

```
<canvas id="canvas" width="200" height="200"></canvas>
```

podstawowe użycie

początkowo płótno jest puste, pozbawione tła. Aby coś na nim wyświetlić, skrypt musi posiadać dostęp do kontekstu renderowania, a następnie musi coś w nim narysować.

canvas posiada metodę getContext() niezbędną, aby uzyskać dostęp do nowego kontekstu renderowania oraz jego funkcji rysujących.

metoda getContext() wymaga podania jednego argumentu, który określa typ kontekstu. Dla rysunków dwuwymiarowych będzie to 2d

```
const canvas = document.getElementById('canvas');
const ctx = canvas.getContext('2d');
```

rysowanie prostokątów

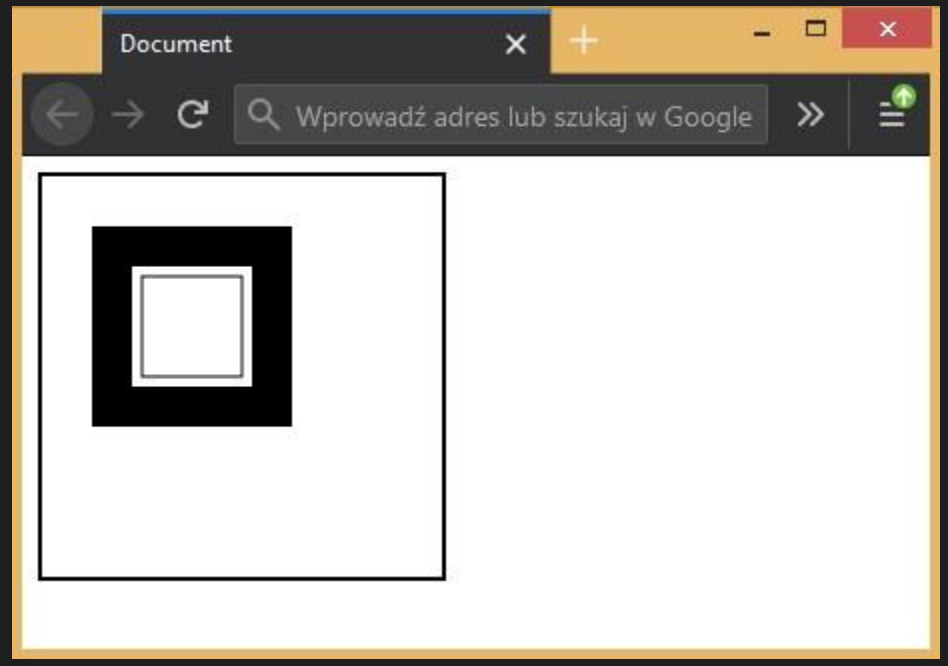
istnieją trzy funkcje rysujące prostokąt:

- fillRect(x, y, width, height) – rysuje pole prostokąta
- strokeRect(x, y, width, height) – rysuje obwód prostokąta
- clearRect(x, y, width, height) – czyści obszar prostokąta

x i y to współrzędne na siatce wyznaczające pozycję lewego górnego rogu prostokąta. Width i height określają jego rozmiar.

rysowanie prostokątów

```
const canvas = document.getElement
ById('canvas');
const ctx = canvas.getContext('2d');

function draw() {
  ctx.fillRect(25, 25, 100, 100);
  ctx.clearRect(45, 45, 60, 60);
  ctx.strokeRect(50, 50, 50, 50);
}
```



wypełnianie elementów kolorem

do wypełnienia elementów można użyć dwóch funkcji

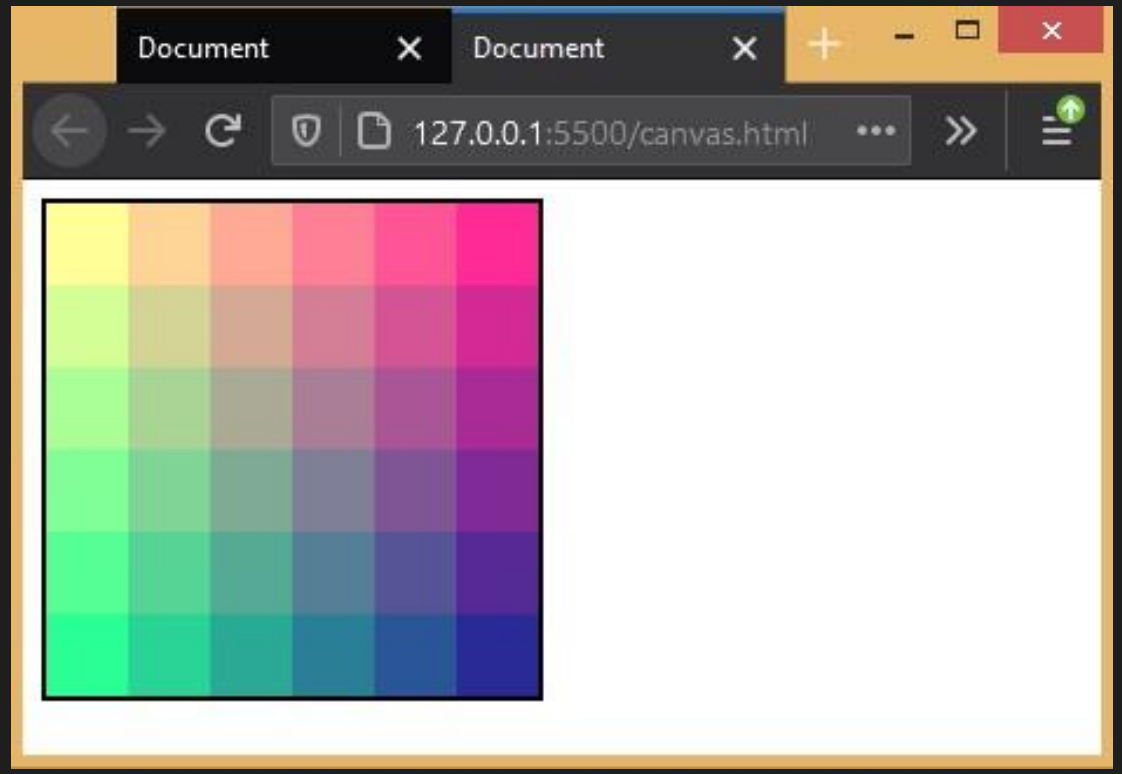
- fillStyle = color – ustawia kolor do wypełnienia kształtów
- strokeStyle = color – ustawia kolor obramowania

opcje określania koloru:

- ctx.fillStyle = 'orange';
- ctx.fillStyle = '#FFA500';
- ctx.fillStyle = 'rgb(255, 165, 0)';
- ctx.fillStyle = 'rgba(255, 165, 0, 1)';

przykład użycia

```
const canvas = document.getElementById
('canvas');
const ctx = canvas.getContext('2d');

function draw() {
    for (let col = 0; col < 6; col++) {
        for (let row = 0; row < 6; row++) {

            ctx.fillStyle = `rgb( ${Math.floor
            (255 - 42.5 * col)}, ${Math.floor
            (255 - 42.5 * row)}, 150)`

            ctx.fillRect(row * 35, col * 35,
            35, 35);
        }
    }
}
```



bibliografia

Książki:

- HTML i CSS. Zaprojektuj i zbuduj witrynę WWW. Podręcznik Front-End Developera, Jon Duckett
- JavaScript i jQuery. Interaktywne strony WWW dla każdego. Podręcznik Front-End Developera, Jon Duckett
- CSS. Kaskadowe arkusze stylów. Przewodnik encyklopedyczny. Wydanie IV, Eric A. Meyer, Estelle Weyl
- UXUI. Design Zoptymalizowany. Manual Book, Chris Badura
- JavaScript. Interaktywne aplikacje webowe, Tomasz Sochacki
- Web Design z HTML5 i CSS3. Technologie frontendowe od podstaw, Terry Felke-Morris

Strony internetowe:

- https://developer.mozilla.org/pl/
- https://css-tricks.com/