WWW

Mateusz Jarosz
mateusja@agh.edu.pl

HTML

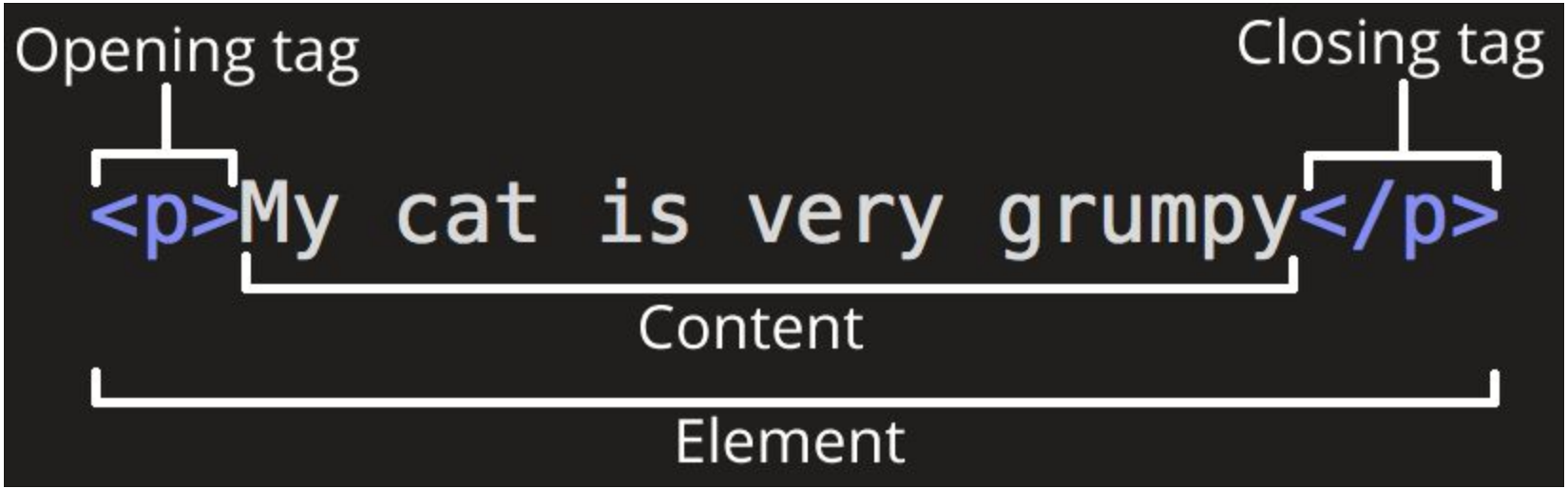
hipertekstowy język znaczników, wykorzystywany do tworzenia dokumentów hipertekstowych.

1980 - Tim Berners-Lee fizyk cern

HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My test page</title>
  </head>
  <body>
    <img src="images/firefox-icon.png" alt="My test image">
  </body>
</html>
```

HTML



https://developer.mozilla.org/pl/docs/Learn/Getting_started_with_the_web/HTML_basics

Podstawowe tagi head

<meta/> nagłówek z metadanymi strony jak autor, tematyka, kodowanie znaków:

<meta name="Author" content="MJ" />

<meta name="Keywords" content="WWW;test" />

<meta name="Description" content="Strona poświęcoana zajęciom www" />

<meta http-equiv="Content-type" content="text/html; charset=UTF-8" />

tytuł <title>strona testowa</title>

logo <link rel="icon" type="image/x-icon" href="/images/favicon.ico">

arkusze stylów <link rel="stylesheet" href="mystyle.css">

Podstawowe tagi body

paragraf

nagłówki <h1> </h1> największy do h6 najmniejszy

pogrubienie

kursywa

odnośniki tekst

zdjęcia <img src=”url obrazka”>

enter

Podstawowe tagi body 2

pozioma linia <hr>

preformatowany tekst <pre> </pre>

dolny / górny index

zaznaczenie <mark> </mark>

podkreślenie <ins> </ins>

przekreślenie <del></del>

Podstawowe tagi body 3

Listy punktowane i numerowane

```
<ul>

    <li> </li>

</ul>

<ol>

    <li> </li>

</ol>
```

lista opisowa

```
<dl>
    <dt></dt>
    <dd></dd>
</dl>
```

Podstawowe tagi body 4

tabele

```
<table>
        <tr>
                <th>kolumna 1</th>          <th>kolumna 2</th>
        </tr>
        <tr>
                <td>wiersz 1 kolumna 1</td>      <td> wiersz 1 kolumna 2</td>
          </tr>
        <tr>
                <td colspan="2">wiersz 1 scalona</td>
          </tr>
</table>
```

Podstawowe tagi body 5

cytowania

<blockquote></blockquote>

<q></q>

<address><address>

<cite></cite>

<bdo><bdo>

Podstawowe tagi body 6

istotne atrybuty

style=”color:red;”

title=”jakiś podpis”

id=”jakieś id” class=”jakaś klasa”

komentarze

<!-- komentarz -->

Bloki

HTML5

HTML5(2014/ 5.1 - 2016) poza dodaniem nowych elementów, usprawniających tworzenie serwisów i aplikacji internetowych, doprecyzowuje wiele niejasności w specyfikacji HTML 4, dotyczących przede wszystkim sposobu obsługi błędów. Niejasności co do sposobu, w jaki przeglądarki powinny obsługiwać błędy w kodzie HTML są jedną z podstawowych przyczyn, dla której wiele serwisów internetowych, napisanych z naruszeniem specyfikacji, w różnych przeglądarkach działa w inny sposób – w niektórych działając, w innych nie. Dzięki HTML-owi 5 obsługa błędów ma być ta sama we wszystkich przeglądarkach, czyli *zły* element będzie działać w każdej przeglądarce albo żadnej.

HTML5 także stawia na semantykę. Element traci na znaczeniu na rzecz <header> <main> <article> <aside> <footer> <nav>, a dodane zostają m.in. <canvas> <figure> <details> <summary>. Element ma być mniej używany na rzecz <mark> <output> <var> <u> <s> (ostatnie dwa znaczniki były w HTML4 przestarzałe – *deprecated*).

HTML5

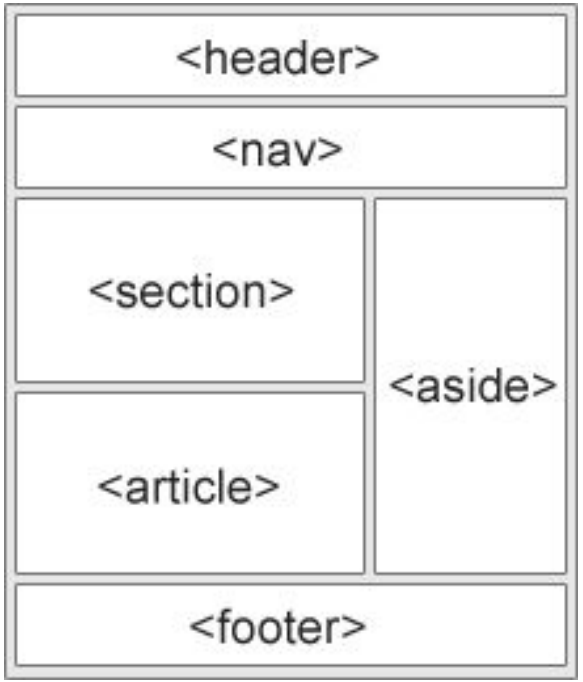
- **Nowe tagi:** section, article, header, footer, nav, video, audio, mark, progress, ...
- **Nowe typy pól "input":** tel, search, url, email, datetime, date, month, week, time, datetime-local, number, range, color.
- **Nowe atrybuty elementów formularzy:** autofocus, required, autocomplete, min, max, multiple, pattern, step, ...
- W poprzednich wersjach HTML, tag <meta> nie miał atrybutu charset, który definiuje standardowe kodowanie znaków dla strony internetowej. Ten atrybut został dodany w HTML5.
- Możliwość osadzenia MathML i SVG bezpośrednio w dokumencie, zupełnie jak w XHTML
- HTML 5 nie zawiera żadnych elementów prezentacyjnych
- HTML 5 nie jest podzielone na żadne tryby – nie ma elementów przestarzałych.

Układ strony

tabele - zamierzchła przeszłość

div - niezalecane wraz z przejściem na html5

section, nav, article, header, footer, aside



CSS

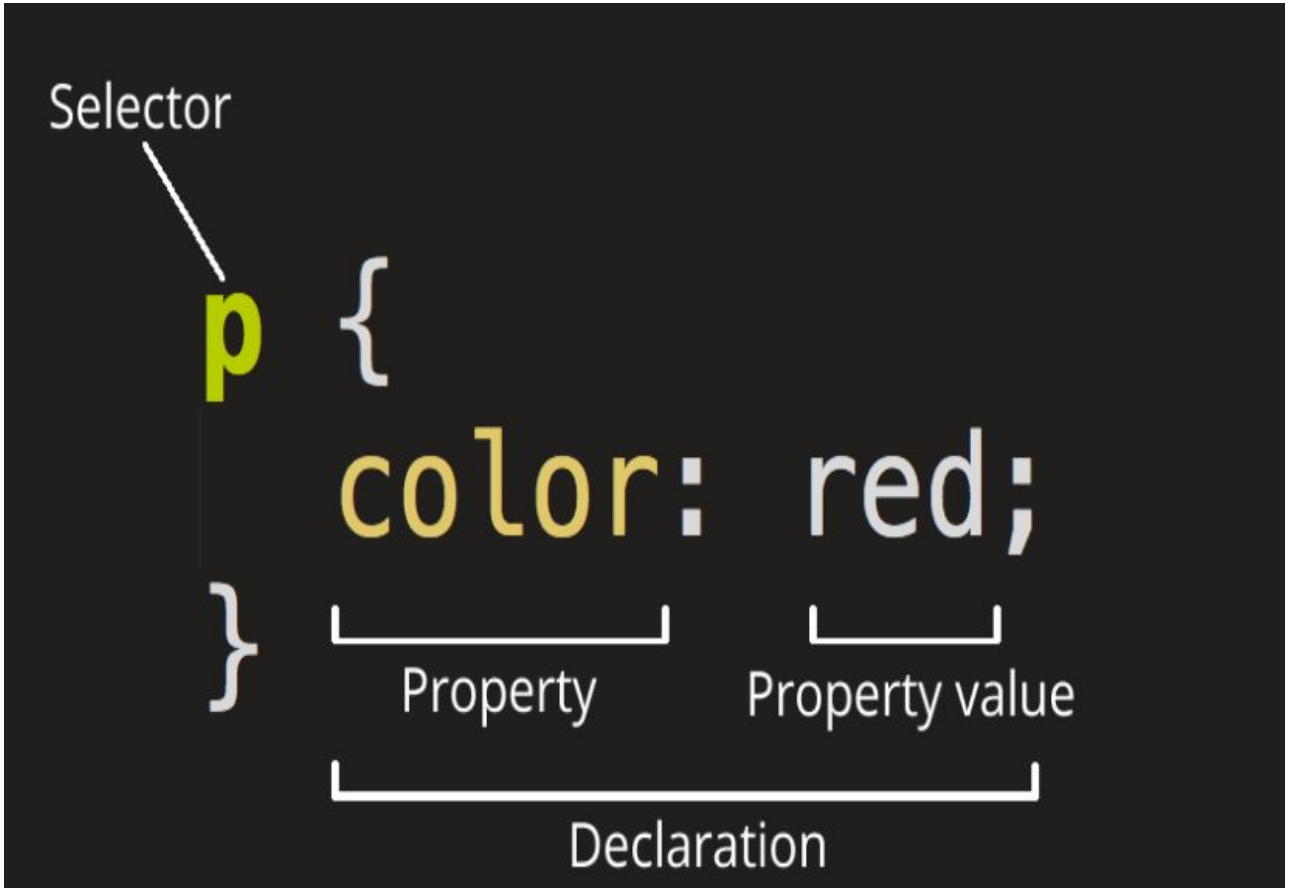
Kaskadowe arkusze stylów CSS (ang. Cascading Style Sheets) to kod służący do nadawania wyglądu strony.

można dodawać go jako inline w argumecie style

w pliku html między tagami <style> w head

jako osobny plik css w tagu <link> w head

CSS



CSS - rodzaje selektorów

- Selektor elementu (czasem nazywany selektorem tagu lub typu) np. p
- Selektor ID np. #my-ID
- Selektor klasy np. .my-class
- Selektor atrybutu np. img[src] dotyczy <img src="myimage.png"> a nie <img>
- Selektor pseudo-klasy np. a:hover Wybiera <a>, ale tylko gdy kursor znajduje się nad elementem.

CSS - podstawowe opcje

color:

background-color:

float:

width:

height:

background:

padding:

margin:
border:
border-radius:
cursor:
display:
visibility:
text-align:
position:
text-decoration:

- a:link
- a:visited
- a:hover
- a:active

przykłady

```
<p style="color:red;">A red paragraph.</p>

<style>
    body {background-color: powderblue;}
    h1   {color: blue;}
    p    {color: red;}
</style>
```

przykłady cd.

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

```
body {background-color: powderblue;}

 h1   {color: blue;}

p     {color: red;}
```

przykłady cd.

```
nav {
  float: left;
  width: 30%;
  height: 600px;
  background: #ccc;
  padding: 20px;
}
```

```
article {
 float: left;
 padding: 20px;
 width: 70%;
 background-color: #f1f1f1;
}
```

przykłady cd.

```
.clearfix::after {
  content: "";
  clear: both;
  display: table;
}
```

Priorytet

stylle są stosowane zgodnie z priorytetem ogólnie mówiąc od najbardziej szczegółowego do najmniej.

Style użytkownika, Style śródliniowe, Typ mediów,

ważność - !important

precyzja - selektor elementu 1, klasy 10, identyfikator 100,

np: p - prioryet 1

np: #ser p - priorytet 101

JS

JavaScript - jest to skryptowy język, który umożliwia obsługę dynamicznego tworzenia treści na stronie internetowej, kontrolowanie multimediów, animację obrazów i prawie wszystko inne.

- Wykonuje się po stronie klienta (można tak odciążać serwer ale bez przesady)
- Obecnie standard i trudno bez niego się obyć

JS - prosty przykład

```
var para = document.querySelector('p');

para.addEventListener('click', updateName);

function updateName() {
  var name = prompt('Enter a new name');
  para.textContent = 'Player 1: ' + name;
}
```

https://developer.mozilla.org/pl/docs/Learn/JavaScript/Pierwsze_kroki/What_is_JavaScript

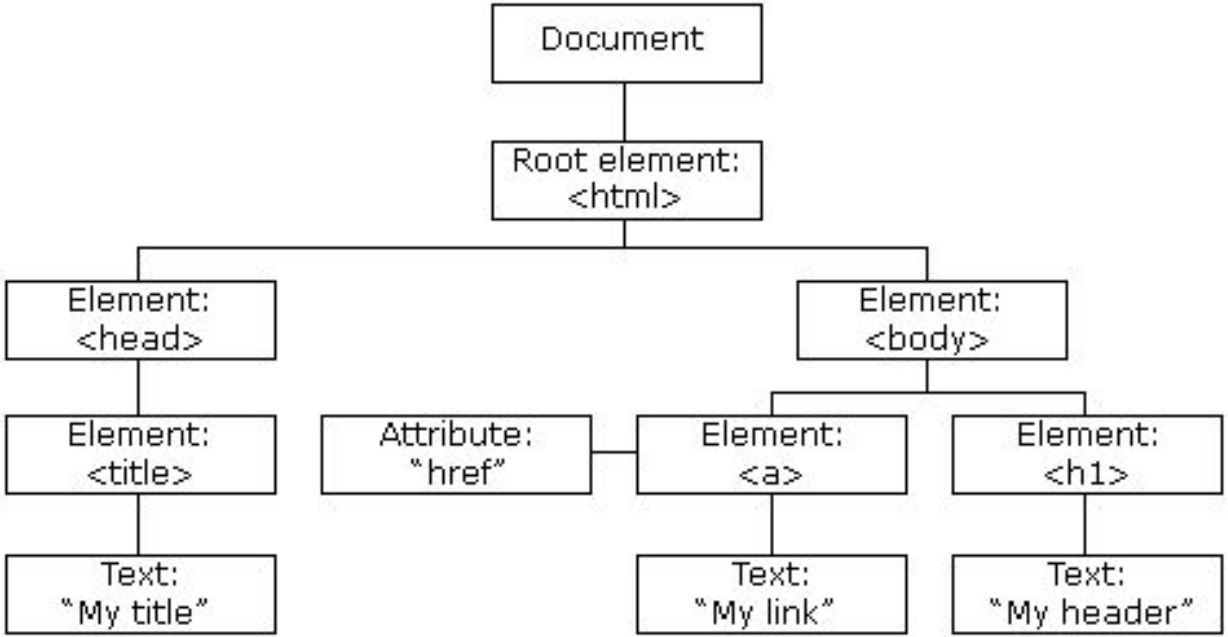
JS - Załączanie skryptu

```
<script>

  // Kod JavaScript będzie tu umieszczony.

</script>
```

```
<script src="script.js"></script>
```

https://developer.mozilla.org/pl/docs/Learn/JavaScript/Pierwsze_kroki/What_is_JavaScript

dom


Document
Root element:
<html>
Element:
<head>
Element:
<body>
Element:
<title>
Attribute:
"href"
Element:
<a>
Element:
<h1>
Text:
"My title"
Text:
"My link"
Text:
"My header"

dom

document.getElementById()

.getElementsByTagName()

.getElementsByClassName()

querySelectorAll() querySelector()

element.innerHTML =

element.*attribute* =

element.style.*property* =

document.createElement()

document.removeChild()

document.appendChild()

document.replaceChild()

document.write()

document.getElementById(*id*).onclick

document.images

document.images[“ID”]

JS - istotne frameworki

- Angular
- React
- Redux

JSP/ERB

JSP (JavaServer Pages) – technologia umożliwiająca tworzenie dynamicznych dokumentów WWW w formatach HTML, XHTML, DHTML oraz XML z wykorzystaniem języka Java, wplecionego w kod HTML danej strony. W tym aspekcie, jest to rozwiązanie podobne do PHP.

ERB (Embedded ruby) podobny do jsp ale dla ruby, kopiuje tekst do wynikowego dokumentu i przetwarza swoje znaczniki

JSP

```
1  <%@ page language="java" contentType="text/html; charset=ISO-8859-2"
2           pageEncoding="ISO-8859-2"%>
3  <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
4            "http://www.w3.org/TR/html4/loose.dtd">
5  <%! int k=5; %>
6  <html>
7    <head>
8      <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-2"/>
9      <title>Przykładowa strona JSP</title>
10   </head>
11   <body>
12      Aktualny czas: <%=java.util.Calendar.getInstance().getTime()%>
13 <%
14       for (int i=0; i<k; ++i) {
15 %>
16         Liczba: <%=i%> <br />
17 <%
18       }
19 %>
20   </body>
21 </html>
```

ERB

<% Ruby code -- inline with output %>

<%= Ruby expression -- replace with result %>

<%# comment -- ignored -- useful in testing %>

% a line of Ruby code -- treated as <% line %> (optional -- see ERB.new)

%% replaced with % if first thing on a line and % processing is used

<%% or %%> -- replace with <% or %> respectively

ERB

```
template = %{
  <html>
    <head><title>Ruby Toys -- <%= @name %></title></head>
    <body>

      <h1><%= @name %> (<%= @code %>)</h1>
      <p><%= @desc %></p>

      <ul>
        <% @features.each do |f| %>
          <li><b><%= f %></b></li>
        <% end %>
      </ul>

      <p>
        <% if @cost < 10 %>
          <b>Only <%= @cost %>!!!</b>
        <% else %>
           Call for a price, today!
        <% end %>
      </p>

    </body>
  </html>
}.gsub(/^  /, '')
```

https://docs.ruby-lang.org/en/2.3.0/ERB.html