Kopienas mājas lapa

Mājas lapas struktūra

Pasākumu finansē: Eiropas Jūrlietu un zivsaimniecības fonds, projekts "Viedo ciemu attīstība piekrastes teritorijās". Nr. 19-00-F043.0443-000001. VRG "Partnerība laukiem un jūrai",sadarbībā ar vadošo partneri VRG "Liepājas rajona partnerība"

NACIONĀLAIS ATTĪSTĪBAS PLĀNS 2020

EIROPAS SAVIENĪBA
Eiropas Jūrlietu un zivsaimniecības fonds

Atbalsta Zemkopības ministrija un Lauku atbalsta dienests

Mēs zinām

- kas ir WEB lapa (HTML, CSS, JS)
- kā glabā un publicē versionētu WEB lapu
- funkcinoālas/nefunkcionālās prasības
- API tehnoloģiju izmantošana mājas lapas veidošanā

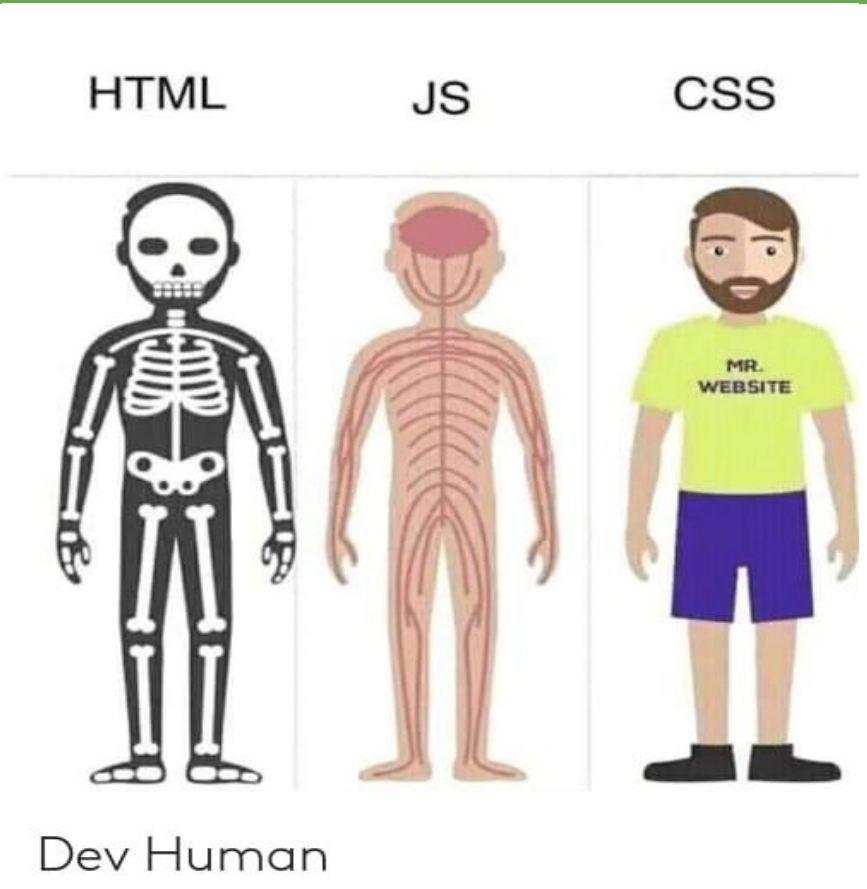
Esam izveidojuši kopienas WEB lapas piedāvājumu, pirmā iterācija.

Šodien uzzināsim

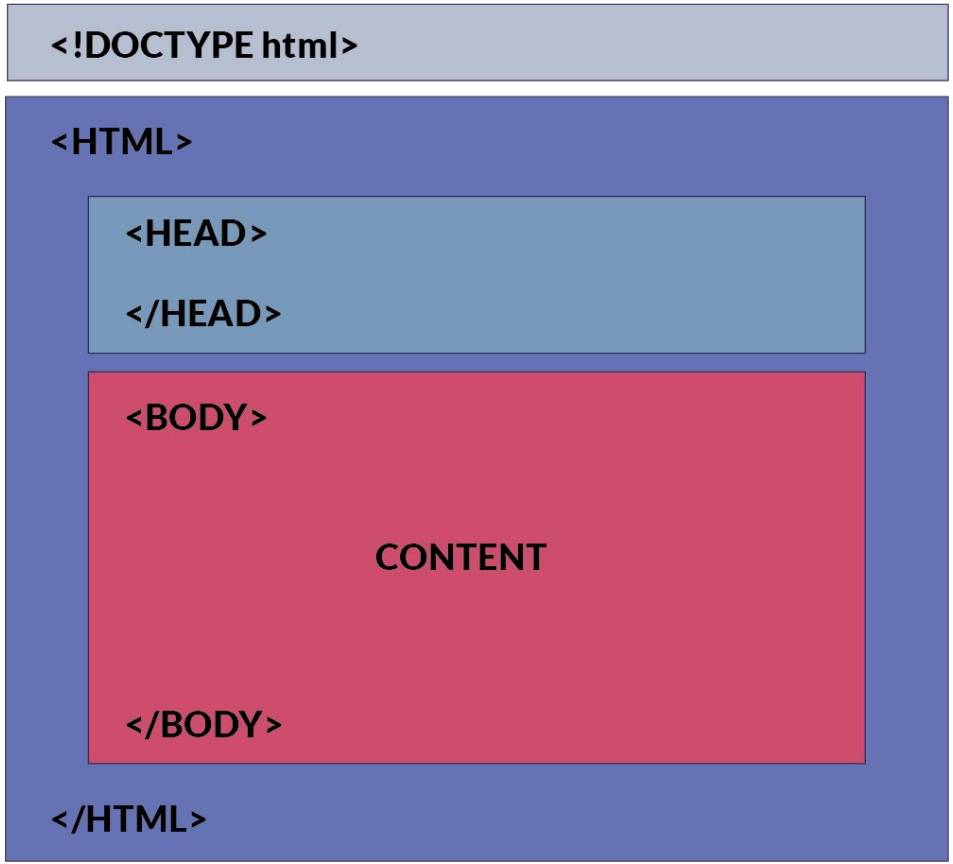
- HTML sintakses pamati
- JS sintakses pamati
- CSS sintakses pamati

Ideatons par kopienas WEB lapu

Nostiprinām pamatus

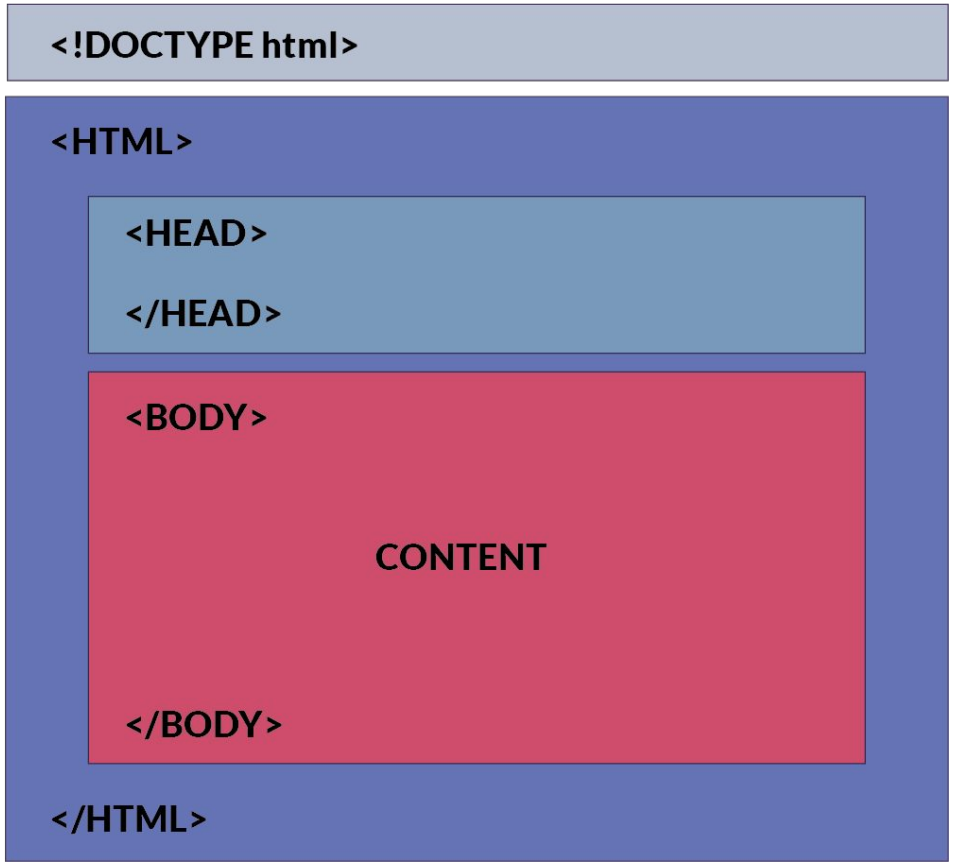


HTMLstruktūra




```
1       <!DOCTYPE html>
2
3       <html lang="en">
4       <head>
5       <meta charset="UTF-8" />
6
7       <title>HTML struktūra</title>
8       <meta name="author" content="Es" />
9       <meta name="description" content="HTML" />
10
11      <link href="style.css" rel="stylesheet" />
12
13      <head>
14      <body>
15      <!-- CONTENT -->
16      </body>
17      </html>
```

TAG - tags



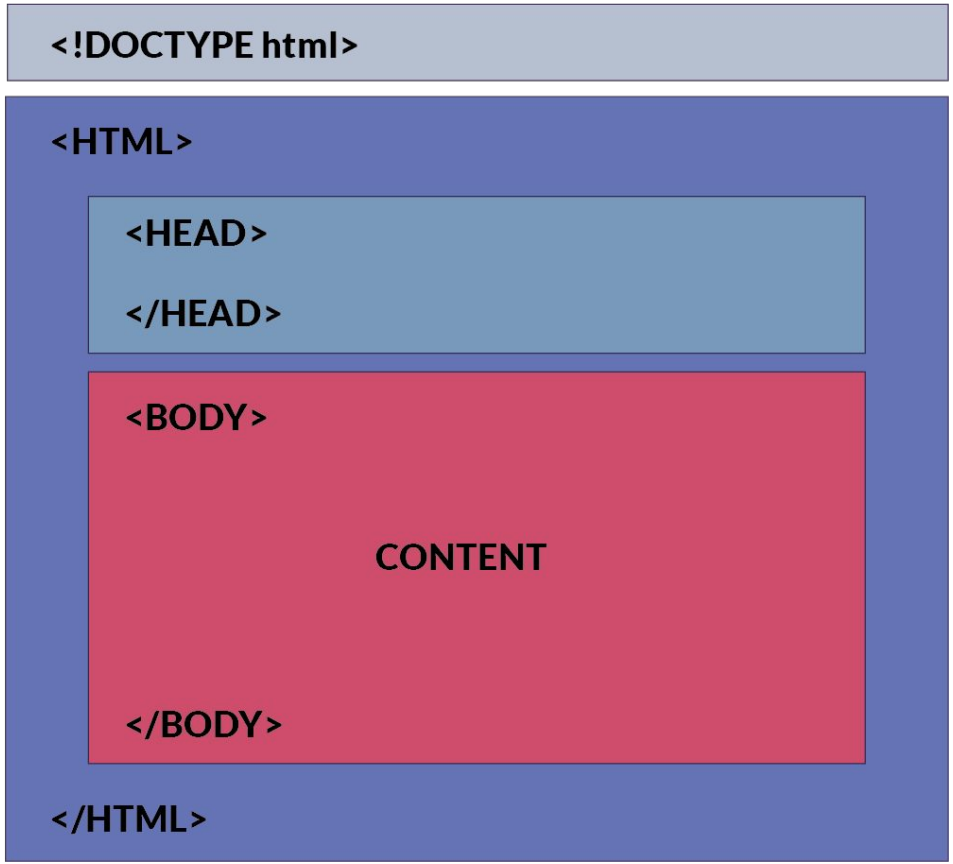
Kas ir tag (eng)?

1. birka
2. etiķete

HTML tagi ir kā atslēgvārdi, kas nosaka, kā tīmekļa pārlūkprogramma formatēs un parādīs saturu. Ar tagu palīdzību tīmekļa pārlūkprogramma var atšķirt HTML saturu no vienkārša satura. HTML tagos ir trīs galvenās daļas: sākuma tags, saturs un beigu tags. Bet daži HTML tagi ir neslēgti tagi.

```
<h1> Virsraksts </h1>
```

DOCTYPE un komentārs



Visiem HTML dokumentiem jāsākas ar `<! DOCTYPE>` deklarāciju.

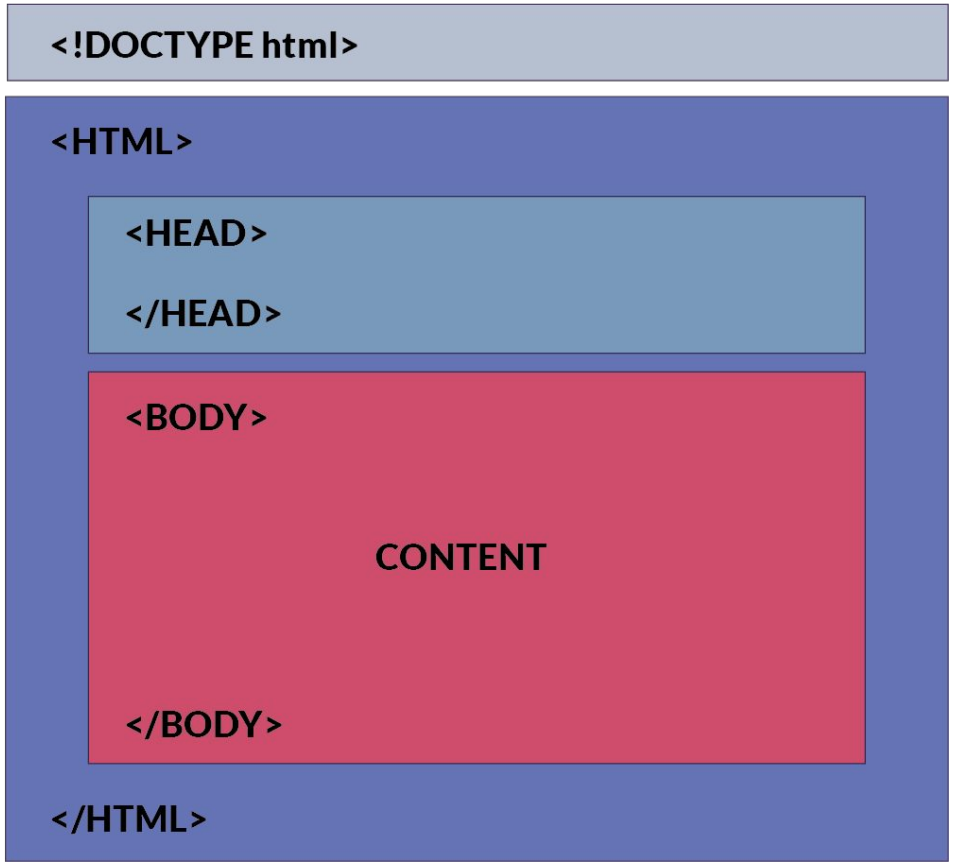
Deklarācija nav HTML tags. Tā ir pārlūka "informācija" par to, kāda veida dokumentu var sagaidīt.

```
<!DOCTYPE html>
```

Komentārs HTML dokumentā:

```
// te ir mans vienas rindas komentārs

<!-- šis ir mans vairāku rindu komentārs
ko es gribu te atrast -->
```

tags - HTML



`<html>` tags ir HTML dokumenta pamats un kalpo kā konteiners visiem pārējiem HTML elementiem.
DOCTYPE nav HTML tags un tapēc tas nav tagā `<html>`

```
<html lang="lv"> … </html>
```

Kas ir tā “lang =”lv” lieta tagā <html>?
Jūs norādāt valodu, kurā tiek rakstīts jūsu saturs. Ja rakstāt vietni, kurā saturs galvenokārt ir rakstīts vācu valodā, jūsu valodas atribūts varētu izskatīties šādi:

HEAD daļa

```
<head>

<meta charset="UTF-8" />
<meta name="author" content="Daina" />
<meta name="description" content="raksta"/>

<title>HTML basic structure</title>

<link href="style.css" rel="stylesheet" />
<link rel="icon" href="images/logo_small.png">

</head>
```

`<head>` tagā ir vispārīga informācija par HTML dokumentu, kā arī saites un skriptu atsauces.

Katrā HTML dokumentā ir jāizmanto tags `<head>`, un tas ir jānovieto, sākot tieši pēc sākuma <html> taga un beidzot tieši pirms sākuma taga `<body>`.

`<meta>` tagā atribūts `charset` (zīmju kodējums) nosaka lapas zīmju kodējumu. UTF-8 standartā. Atribūts `name` nosaka informācijas nosaukumu un nāk komplektā ar `content` (saturs).

.... HEAD daļa

```
<head>

...

<title>HTML basic structure</title>
<link href="style.css" rel="stylesheet" />
<link rel="icon" href="images/logo_small.png">

</head>
```

`<title>` tagā rakstam lapas nosaukumu.
To izmantos:

- pārlūka rīkjoslā;
- kā virsrakstu, pievienojot lapu izlasei;
- kā lapas nosaukumu, kad tas tiek parādīts meklētājprogrammas rezultātos.

`<head>` daļā tagu `<link>` izmanto, lai “saistītu” ārējos resursus ar mūsu HTML dokumentu. Tās galvenais mērķis ir saistīt ar ārēju stila lapu (CSS). Šajā tagā `<link>` ir divi galvenie atribūti:

```
rel = ”stilu lapa”
href = ”XXX.css”
```

Atribūtā “rel” mēs paziņojam, uz ko saite attiecas (stila lapa). “href” mēs dokumentam norādām, kur atrodas šī ārējā stila lapa (ceļš uz CSS failu). “href” nozīmē “Hipersaites atsauce”.

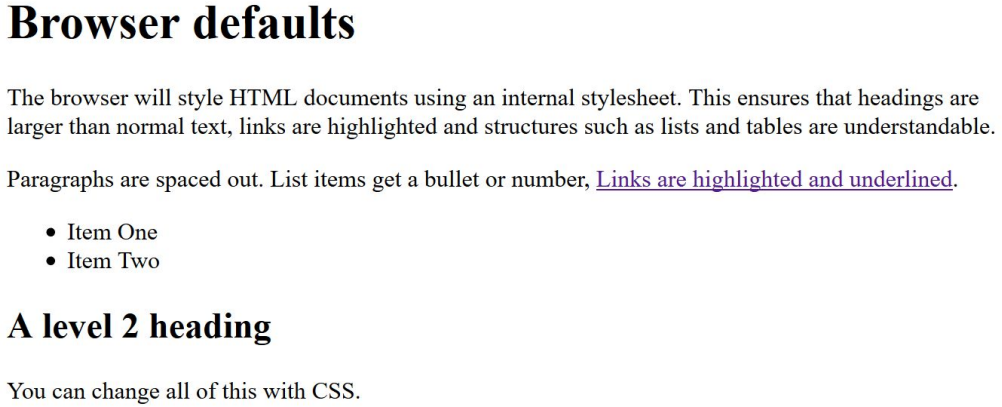
BODY daļa

```
<body>

…
…
…
…


</body>
```

Tātad, mēs esam tuvu sava koda beigām. Pēdējais ir tags `<body>`.
Tagā `<body>` ir mūsu HTML dokumenta pamatteksts. Tajā ir visi elementi, kas veido mūsu HTML dokumenta saturu.
Tātad viss, kas atrodas tagā <body>, tiks parādīts kā saturs mūsu pārlūkprogrammā.

Te var būt dažādi tagi un viņu vērtība `<div>, <h1>, <p>, <a>, <i>, <header>, <span>, <ul>, <li>, <br>, <b>, <footer>, ...` .
Vairāk info par katra taga lietošanu un sintaksi te - https://www.w3schools.com/html/html_elements.asp

CSS



CSS ir valoda, lai norādītu, kā dokumenti tiek rādīti lietotājiem - kā tie tiek veidoti, izkārtoti utt.

Kreisajā pusē redzams noklusētais pārlūka stils

Tīmeklis būtu garlaicīga vieta, ja visas vietnes izskatītos šādi. Izmantojot CSS, jūs varat precīzi kontrolēt, kā pārlūkprogrammā izskatās HTML elementi, uzrādot marķējumu, izmantojot jebkuru dizainu, kas jums patīk.

Video par CSS un tā vēsturi - https://youtu.be/spK_S0HfzFw

CSS sintakse

```
/* Es vēlos, lai manas lapas
galvenā virsraksts tiktu
parādīts kā liels sarkans
teksts */
h1 {
    color: red;
    font-size: 5em;
  }
```

CSS ir uz noteikumiem balstīta valoda - jūs definējat stilu grupas, kas jāpiemēro noteiktiem jūsu tīmekļa vietnes elementiem vai elementu grupām.

Piemēram, "Es vēlos, lai manas lapas galvenā virsraksts tiktu parādīts kā liels sarkans teksts."

Šis kods parāda ļoti vienkāršu CSS kārtulu, kas nodrošina iepriekš aprakstīto stilu:

CSS sintakse

```
h1 {
    color: red;
    font-size: 5em;
  }
```

CSS ir uz noteikumiem balstīta valoda - jūs definējat stilu grupas, kas jāpiemēro noteiktiem jūsu tīmekļa vietnes elementiem vai elementu grupām.

Piemēram, "Es vēlos, lai manas lapas galvenā virsraksts tiktu parādīts kā liels sarkans teksts."

Šis kods parāda ļoti vienkāršu CSS kārtulu, kas nodrošina iepriekš aprakstīto stilu:

CSS sintakse

```
p, li {
     color: green;
}
li {
  list-style-type: none;
}
```

CSS ir uz noteikumiem balstīta valoda - jūs definējat stilu grupas, kas jāpiemēro noteiktiem jūsu tīmekļa vietnes elementiem vai elementu grupām.

Piemēram, "Es vēlos, lai manas lapas galvenā virsraksts tiktu parādīts kā liels sarkans teksts."

Šis kods parāda ļoti vienkāršu CSS kārtulu, kas nodrošina iepriekš aprakstīto stilu:

CSS izmantojot klasi

```
<ul>
  <li>Item one</li>
  <li class="special">Item two</li>
  <li>Item <em>three</em></li>
</ul>
```

```
li.special,
span.special {
  color: orange;
  font-weight: bold;
  }
```

```
.special {
  color: orange;
  font-weight: bold;
  }
```

CSS izmantojot vietu HTML dokumentā

```
h1 + p {
  font-size: 200%;
}
```

```
li em {
  color: rebeccapurple;
}
```

```
<h1>Virsraksts</h1>
<p>Paragrāfs ar saiti <a href="http://engure.te.lv">link</a>.</p>
<p>Otrs paragrāfs ar <em>emphasized</em> elementu</p>

<ul>
   <li>Vienība <span>pirmā</span></li>
   <li>Otrā vienība</li>
   <li>Vienība <em>trešā</em></li>
</ul>
```

CSS pēc elementa statusa

```
<a href="https://engure.te.lv" >Etech</a>
```

Hipersaite var būt jauna, jau reiz skatīta un tā var mainīt krāsu pārbdraucot ar peli. To visu var izmantot definējot stiu pēc hipersaites statusa.

```
a:link {
    color: pink;
}

a:visited {
    color: green;
}

a:hover {
    text-decoration: none;
}
```

CSS izteiksmes kombinēšana

```
body h1 + p .special {
  color: yellow;
  background-color: black;
  padding: 5px;
}
```

Šis kods piešķirs doto stilu katram elementam ar klasi `special`, kas definēts tagā `<p>`, kurš savukārt nāk uzreiz aiz `<h1>`, Un `<h1>` ir definēts `<body>`.

Juhuuuu!

Pamēģini uzrakstīt HTML kodu, kas ļauj to lietot :)

CSS piemērs

```
/* ikviens <span> kas ir iekš <p>,
kas ir iekš <article>  */
article p span { ... }

<article>
<p>
<span>Es esmu tas kuru maina</span>
</p>
<span>Mani nemaina</span>
</article>
<span>Mani nemaina</span>
```

```
/* ikviens <p> kas nāk uzreiz aiz <ul>,
kas nāk uzreiz aiz <h1>  */
h1 + ul + p { ... }


<h1>
<ul>
<p> Es esmu tas ko maina</p>
<p> Mani nemaina</p>
</ul>
<ul>
<p> Mani nemaina</p>
</ul>
</h1>
```

Laba apmācība ir te - https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps

Java Script

JavaScript (saīsināti "JS") ir pilnvērtīga dinamiskās programmēšanas valoda, kas WEB lapai pievieno interaktivitāti (funkcijas, darbības). To izgudroja Brendans Eihs (Mozilla projekta, Mozilla fonda un Mozilla Corporation līdzdibinātājs).
JavaScript valodai ir pilna funkcionalitāte. Ar lielāku pieredzi jūs varēsiet izveidot spēles, animētas 2D un 3D grafikas, visaptverošas uz datu bāzēm balstītas lietotnes (WEB lapas) un daudz ko citu!

https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/JavaScript_basics

Java Script mainīgie - variables

Mainīgie ir konteineri, kas glabā vērtības. Sāciet, deklarējot mainīgo ar var (mazāk ieteicams) vai let atslēgvārdu, kam seko nosaukums, kuru piešķirat mainīgajam.

```
let myVariable;
```

Java Script mainīgie - variables

```
let myVariable;
```

```
let myName = 'Chris';

myName = 'Bob';
```

Mainīgie ir konteineri, kas glabā vērtības. Sāciet, deklarējot mainīgo ar var (mazāk ieteicams) vai let atslēgvārdu, kam seko nosaukums, kuru piešķirat mainīgajam.

Jūs varat izsaukt mainīgo jebkā, kā jums patīk, taču ir ierobežojumi. Izmantojot tikai latīņu rakstzīmjes (0–9, a – z, A – Z) un pasvītrojuma rakstzīmes.

Atšķirība starp var un let definējot mainīgo
https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/Variables#the_difference_between_var_and_let

Pārbaudīt vai mainīgā nosaukums ir derīgs - https://mothereff.in/js-variables

Java Script mainīgo tipi

```
let myAge = 17;
let dolphinGoodbye = 'So long and thanks for
all the fish';
let iAmAlive = true;
let test = 6 < 3;
let myNameArray = ['Chris', 'Bob', 'Jim'];
let myNumberArray = [10, 15, 40];
myNameArray[0]; // should return 'Chris'
myNumberArray[2]; // should return 40
let dog = { name : 'Spot', breed : 'Dalmatian'
};
dog.name
const daysInWeek = 7;
```

number - skaitlis,
string - rakstzīmju virkne
boolean - patiess, nepatiess (true, false)
arrays - massīvi
objects - objekti

JavaScript ir dinamiska programmēšanas valoda, tāpēc Jums nav iepriekš jāsaka, kāda tipa mainīgais katrs būs. Tas var programmas laikā mainīties.

const - konstante
Glabā informācija kas visas programmas laikā netiks mainīta.

Java Script mainīgo tipi

```
let myAge = 17;
let dolphinGoodbye = 'So long and thanks for
all the fish';
let iAmAlive = true;
let test = 6 < 3;
let myNameArray = ['Chris', 'Bob', 'Jim'];
let myNumberArray = [10, 15, 40];
myNameArray[0]; // should return 'Chris'
myNumberArray[2]; // should return 40
let dog = { name : 'Spot', breed : 'Dalmatian'
};
dog.name
const daysInWeek = 7;
```

number - skaitlis,
string - rakstzīmju virkne
boolean - patiess, nepatiess (true, false)
arrays - massīvi
objects - objekti

JavaScript ir dinamiska programmēšanas valoda, tāpēc Jums nav iepriekš jāsaka, kāda tipa mainīgais katrs būs. Tas var programmas laikā mainīties.

const - konstante
Glabā informācija kas visas programmas laikā netiks mainīta.

Java Script funkcija

```
var x = myFunction(4, 3);   // Function is
called, return value will end up in x

function myFunction(a, b) {
  return a * b;           // Function returns the
product of a and b
}
```

JavaScript funkcija ir koda bloks, kas paredzēts konkrēta uzdevuma veikšanai.

JavaScript funkcija tiek izpildīta, kad "kaut kas" to izsauc (izsauc).

JavaScript funkcija tiek definēta ar funkcijas atslēgvārdu, kam seko nosaukums, kam seko iekavas (). Funkciju nosaukumos var būt burti, cipari, pasvītras un dolāra zīmes (tie paši noteikumi kā mainīgie). Iekavās var būt parametru nosaukumi, kas atdalīti ar komatiem: (parametrs1, parametrs2, ...)
Funkcijas izpildāmais kods ir ievietots figūriekavās: {}

Piemēri

```
function toCelsius(fahrenheit) {
  return (5/9) * (fahrenheit-32);
}
document.getElementById("demo").innerHTML =
toCelsius(77);
```

```
var x = toCelsius(77);
var text = "The temperature is " + x + " Celsius";
```

```
// code here can NOT use carName

function myFunction() {
  var carName = "Volvo";
  // code here CAN use carName
}

// code here can NOT use carName
```

```
function myFunction() {
  alert("Hello World!");
}
```

Apmācības nodrošina NVO “Piekrastes Konvents”. Vairāk info - info@piekrasteskonvents.lv